młody TECHNIK

InforMik

MAGAZYN KOMPUTEROWY „MŁODEGO TECHNIKA"

II
1989



Nasz komentarz

KOŁOPROCESOR A SPRAWA POLSKA

Przypuśćmy, drogi Czytelniku, że zepsuł Ci się telewizor kolorowy. Zaiskrzyło, zaśmierdziało i obraz zniknął. Podejrzewasz zwykłe przepalenie jakiegoś rezystora, ale czas nagli i śpieszysz do warsztatu, aby zlecić naprawę. Telewizor to oczko w głowie całej rodziny, wybierasz więc zakład budzący zaufanie i obstawiony reklamami w rodzaju: „Super-hiper service", „Tylko u nas tanio, szybko i solidnie" lub: „Zatrudniamy najlepszych profesjonalistów" itd. Za kilka dni zjawiasz się w warsztacie po odbiór aparatu, bierzesz do ręki rachunek i nagle czujesz, że tracisz grunt pod nogami: skromne ćwierć miliona czeka na uregulowanie. Z oburzeniem skaczesz do oczu eleganckiemu drabowi w krawacie: „Za co ćwierć bańki? Przecież zapalił się tylko opornik! Musieliście pomylić odbiorniki!". Serwisant nie traci rezonu: „Całkiem możliwe, że to tylko opornik. W takich przypadkach wymieniamy jednak zawsze całą elektronikę wraz z kineskopem. Tylko w ten sposób mamy całkowitą pewność, że TA usterka już się nie powtórzy".

Czy powyższa scenka rodzajowa jest abstrakcyjna? Jeśli chodzi o telewizory, zapewne tak. Gdyby ktoś odważył się naprawdę na taki numer, znalazłby się sposób, aby mu zalać sadła za skórę. Jest przecież cech, sąd, gazety i „Teleexpress". Przede wszystkim jest zaś wiele konkurencyjnych zakładów naprawczych, z których część prowadzą naprawdę kompetentni specjaliści, znający przedmiot swej działalności od podszewki. A jak przedstawia się to w serwisie komputerów? Tym razem scenka nie tylko mogłaby się przydarzyć naprawdę, ale z pewną odmianą powtarza się nieustannie w różnych miejscach naszego pięknego kraju. Odmiana polega na tym, że klient raczej nie protestuje, ale pokornie płaci rachunek. Dlaczego?

Jak wiadomo, lwia część mikrokomputerów w Polsce pochodzi z importu, często chaotycznego, przypadkowego i kierującego się wyłącznie minimalnymi cenami. Rozmaitość sprzętu jest wielka, dostępność aktualnej dokumentacji technicznej — prawie żadna. Z nielicznymi wyjątkami brak autoryzowanego, dobrze wyposażonego serwisu, reprezentującego zachodnich producentów. Mnóstwo nowo powstałych firm handlowych chcąc egzystować MUSI zapewnić JAKIŚ serwis klientom (przynajmniej gwarancyjny), gdyż w przeciwnym razie tych klientów po prostu może nie być. Wobec powyższego w pakamerze na zapleczu organizuje się „pracownię", zatrudniając w niej ze dwóch techników z ogłoszenia w gazecie.

Wystarczy przyjrzeć się w takich firmach serwisowi komputerów „od kuchni", aby raz na zawsze stracić ochotę do korzystania i ich usług. Poszukiwanie usterek polega na przekładaniu pakietów i kombinowaniu, w jakiej konfiguracji komputer funkcjonuje. Mierniki, jeśli są używane, służą do sprawdzania napięcia sieci. Po oscyloskop nikt raczej nie sięga, gdyż i tak nie bardzo wiedziałby, co i gdzie można zaobserwować. No, może ewentualnie falę prostokątną na wyprowadzeniach łatwego do odróżnienia rezonatora kwarcowego... Jakkolwiek usterka płyty głównej, jeżeli nie spowodowała jej łatwa do wykrycia awaria układu pamięci RAM, kończy się z reguły wymianą tej płyty. To samo dotyczy innych kart. Co zmyślniejszy serwisant poradzi sobie jeszcze z awarią układów separujących w interfejsach szeregowych i równoległych, choć tutaj nierzadko wymiana odbywa się „na pałę". Rytualne przekładanie kart w gniazdach oraz kolejne wyjmowanie i wkładanie układów w podstawki przypomina żywo obrządek dla odpędzenia złych duchów...

O konkurencji trudno mówić, skoro większość firm działa podobnymi metodami — dotyczy to nawet tych, które mile łechcą nasze oczy wielkimi reklamami w czasopismach lub efektownymi wstawkami w telewizji. Życie intymne mikrokomputera pozostaje czarną magią, nic więc dziwnego, że zamiast lekarzy praktykują nieraz komputerowi znachorzy i szarlatani, trzymający jednak niezawodnie fałszywy, profesjonalny szpan.

Kwalifikacje wielu serwisantów są raczej manualne niż inżynierskie. Autoryzowanych przez producenta, a przynajmniej zorganizowanych na przyzwoitym poziomie form doskonalenia kwalifikacji raczej brak, do samodzielnego samokształcenia brak odpowiedniej literatury i dokumentacji, a czasem także czasu i chęci. Zresztą po co studiować schematy, skoro po trochu trzeba zajmować się wszystkim, a każdy nowy pakiet może okazać się jednodniową znajomością. W cenie jest refleks i intuicja połączona ze sporą dawką tupetu. Serwisista staje się narzędziem napędzającym sprzedaż i temu celowi powinny służyć jego poczynania.

Pamiętam, jak ongiś warszawska firma sprzedawała komputery katowickiej kopalni, wyposażając jeden z nich w koprocesor arytmetyczny. Ponieważ klient z koprocesora zrezygnował, firma wysłała swego emisariusza, aby cenną kostkę wyjął. Serwisista zapewne widział już kiedyś komputer od środka, gdyż przypomniał sobie, że koprocesor to duży „robaczek" o 40 nóżkach, umieszczony koło procesora (bywa tak w komputerach klasy XT). Zręcznym ruchem wyjął więc kostkę i powrócił do stolicy w poczuciu dobrze wykonanej roboty. Złośliwy pech sprawił jednak, że komputer był klasy AT, a koprocesor znajdował się całkiem gdzie indziej. Dzielny fachowiec wydłubał z komputera „kołoprocesor", tzn. sterownik przerwań — w końcu też „kostka" o 40 końcówkach i umieszczona tak blisko procesora...

Serwis sprzętowy jest w wielu firmach smutną koniecznością, mało kto jest poważnie zainteresowany inwestowaniem w tym kierunku. Zwłaszcza że najpoważniejszą inwestycję stanowią ludzie, którzy po nabyciu kwalifikacji zawsze mogą „ulotnić się" do konkurencji. Co operatywniejsi handlowcy urządzają się chytrzej. W okresie gwarancyjnym przesyłają uszkodzone podzespoły do ich zachodnich dostawców, którzy w końcu także udzielają gwarancji dla swych dostaw. Gdy gwarancja skończy się, można wszelkie wydatki przerzucić na klienta, który pokryje koszty nowego mainboardu nawet wtedy, gdy usterkę spowodowało zwykłe pęknięcie ścieżki drukowanej.

To, co napisałem powyżej, to nie zarzuty pod adresem małych częstokroć, lecz operatywnych firm, lecz opis rzeczywistości, kształtującej się pod wpływem obiektywnych okoliczności. Mamy usługową nadbudowę, lecz brakuje produkcyjnej bazy. Przemysł (mikro) komputerowy w Polsce przypomina drzewo o bujnych gałęziach, ale pozbawione pnia i unoszące się nad ziemią w odległości zmieniającej się w takt modyfikacji przepisów celno-skarbowych. Owa ziemia-baza to wielcy producenci podzespołów, pakietów, jednostek pamięci, dyskietek, itd. Między konarami drzewa uwijają się skrzętne krasnoludki, które montują, instalują, kompletują, wymieniają, ale nie pomnażają ogólnej masy dóbr. Masa ta napływa z zewnątrz. Energia mieszkańców drzewa skupia się więc na pokonywaniu przestrzeni między drzewem a ziemią i sztucznym zasilaniu drzewa w krzemowe soki, tak by nie uschło.

Tam, gdzie dostawcą soków jest własny przemysł wytwórczy z odpowiednim zapleczem, usługi nie działają w pustce, między produkcją a usługami trwa ciągły przepływ materiałów, ludzi i know-how. Profesjonaliści mogą pogłębiać swe kwalifikacje bez „partyzantki" i nadzwyczajnych wyrzeczeń, w sposób naturalny zmieniając stanowiska pracy. Drobny handel ma oparcie w hurcie i organizacjach serwisowych z prawdziwego zdarzenia, ze zorganizowanymi dostawami części zamiennych i wsparciem ze strony producentów. W Polsce brak nam pnia i korzeni, tej luki nie wypełnią ruchliwe i skądinąd bardzo potrzebne „drobnoustroje" w postaci rozmaitych spółek i zakładów rzemieślniczych.

Miałem okazję obejrzeć na targach CeBIT w Hanowerze bogatą ekspozycję indyjską, co wprawiło mnie w przygnębienie. Polscy inżynierowie nie ustępują indyjskim. W Polsce (mikro) komputeryzację oddano na pastwę żywiołowej przedsiębiorczości obywateli, rozmieniając ją na barwną mozaikę firm i firmek, ale nie tworząc bazy. Baza wymaga zorganizowanych działań na poważną skalę. Poważne instytucje troszczą się jednak o węgiel i żelazne kęsiska, a nie o byle drobiazgi o składzie chemicznym tak podobnym do piasku... Tymczasem w Indiach, uważanych przez wielu za modelowy kraj Trzeciego Świata, udało się, co prawda z obcą pomocą, szybko uruchomić przemysł komputerowy. To nic, że na razie montuje się tam głównie urządzenia zaprojektowane gdzie indziej. Przepływ myśli technicznej już się zaczął i zapewne będzie przybierać na sile. Polska centrala METRONEX, reprezentująca polską bazę, wystawiła w Hanowerze legendarną już drukarkę D-100 i mechaniczne maszyny do pisania. Wymarzony ekwipunek dla indyjskiego fakira...

Roland Wacławek

młody TECHNIK
InforMik
MAGAZYN KOMPUTEROWY
MŁODEGO TECHNIKA
NR II [10] ROCZNIK III

SPIS TREŚCI

FELIETONY:
KOŁOPROCESOR A SPRAWA POLSKA — Roland Wacławek . . II s. okł.
ELEKTRONICZNY SŁUGA — Jerzy Klawiński . 1
ARTYKUŁY:
OBSŁUGA PRZERWAŃ ASYNCHRONICZNYCH W TURBO-PASCALU 4.0 I 5.0 — Roland Wacławek . 2
CEBIT PO RAZ CZWARTY — Roland Wacławek . 6
NAJPROSTSZY RAM-DYSK DLA ZX SPECTRUM — Grzegorz Zalot . 11
C-64 JAKO MIERNIK CZĘSTOTLIWOŚCI — Wojciech Żurek 18
TRANSMITUJMY! — Janusz Wrześniak 24
OPTYCZNE PAMIĘCI DYSKOWE — Krzysztof Wiśniewski 28
STAŁE DZIAŁY:
KOMPUTER W SZKOLE — opr. Ireneusz Włodarczyk 30
CIEKAWE KSIĄŻKI . . . . 26 i III s. okł.
RÓŻNE:
CO NOWEGO Z AMIGĄ? 27

Zdjęcia w numerze: T. Basista, W. P. Jabłoński, S. Rutkowski, G. Zalot, „Tool", „Byte".
Numer ilustrował: Roman Gaik.

ELEKTRONICZNY SŁUGA

...Stoi sobie na stole i nic! Nie przemawia jeszcze ludzkim głosem, nie aktywizuje się samoczynnie w stosownych porach, a o niektórych jego możliwościach nie wie często nawet zapalony użytkownik. A jednak niepokoi...

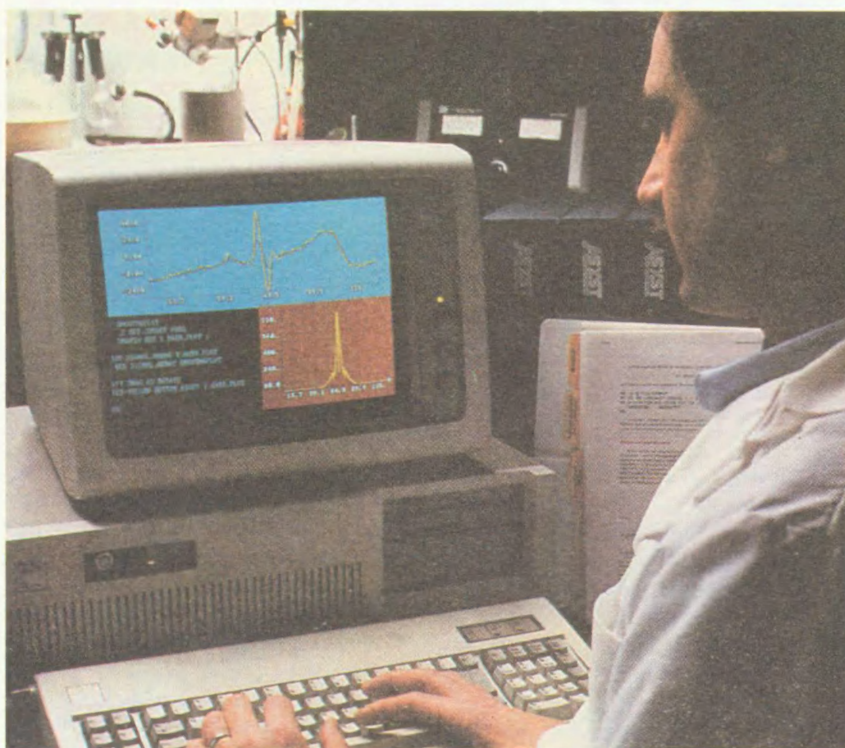
Wiele maszyn — prostszych i bardziej skomplikowanych — towarzyszy człowiekowi od zarania dziejów. Mamy do nich najczęściej dość ambiwalentny stosunek — potrzebujemy ich w pracy i domu, a jednocześnie narzekamy na ich toporność, wrażliwość na niewłaściwe traktowanie przez użytkowników, brak umiejętności samodzielnego zrobienia za nas tego, co się nam przyśni. Komputer jednak jest partnerem bardziej wymagającym od sochy czy traktora, potrafi niewprawnego użytkownika ochrzanić głośnym buczeniem, czy odpowiednio krytyczną (choć grzeczną) inskrypcją na ekranie. Dlatego — być może — wiele osób owa komputerowa szczerość zniechęca. Przypomina mi to zachowanie pewnej byłej (czy może w złych czasach utajnionej?) hrabiny, która na jakąś celną i ostrą uwagę miała powiedzieć: „Co z tego, że to prawda, ale jaka oburzająco nietaktowna!" I cóż z tego, że komputer ma rację, jeśli poucza swego właściciela o jego nieuctwie! To oburzające! Dawni lokaje byli stanowczo lepiej wychowani!

Żyjemy w czasach, gdy wiele się mówi o służbie (ludzkości, szczytnym ideałom humanizmu, narodowi itp., itd.), poszukuje się służby („Młodą, bez nałogów, uczciwą o dobrej prezencji..."), ale myśli się o niej z niejaką pogardą („Nie jestem twoją służącą!!!"). Serwilizm komputera jest natomiast niezwykle specyficzny: służy on wiernie i sprawnie (choć bez uniżoności!) tylko ludziom mądrym — przynajmniej na płaszczyźnie informatyki. Co więcej — spełnia wszelkie polecenia cicho, dyskretnie i szybko, choć niestety nie ma uroku czarnej pokojówki z serialu „W kamiennym kręgu" mimo opływowej linii klawiatury, monitora i płyty głównej. Ma natomiast kolosalną wadę — jego władca musi trzymać się określonych reguł i nie ma szans na wykonanie głupich poleceń.

Fantastyka naukowa nauczyła nas bać się elektronicznej służby: po kartach wielu książek szaleją tłumy zbuntowanych przeciw ludzkiej głupocie robotów, stworzonych w zamyśle dla służenia człowiekowi. Natomiast życie uczy nas cenienia mechanizmów użytecznych na co dzień i od święta. Czy komputer stanie się u nas tak popularny jak mlynkomikser czy maszynka do mięsa? Chyba tak, choć — moim skromnym zdaniem — dopiero w następnym pokoleniu. Skąd taki wniosek? Ano stąd, że nasi dziadkowie golili się wyłącznie niemal za pomocą brzytwy, gardząc maminsynkami golącymi się maszynką i żyletką, a na golarkę elektryczną patrzeć wprost nie mogli. Co innego nasi ojcowie — ci nawet dla samego snobizmu golili się „godzącymi w ..." maszynkami „Remington" lub „zgodnymi z..." firmy „Tuła". To samo będzie i z komputerami — trzeba się z nimi po prostu oswoić.

Czy grozi nam kiedyś bunt „szybkiego wariata"? Może tak, może nie. Przecież bardzo wiele zależy od tego, czym będą go karmić programiści. Na razie nasz elektroniczny sługa minął rafy mody i wpłynął na monotonne fale oceanu codzienności. Może zmierza ku nie znanemu dotąd lądowi przyjaźni pomiędzy człowiekiem i maszyną?

JERZY KLAWIŃSKI



ROLAND WACŁAWEK

OBSŁUGA PRZERWAŃ ASYNCHRONICZNYCH W TURBO-PASCALU 4.0 i 5.0

Do niedawna do odparcia ataku przeciwników Asemblera wystarczały niezawodne argumenty: programowanie obsługi przerwań, zwłaszcza tych asynchronicznych, a więc wywoływanych sprzętowo w chwili niemożliwej do przewidzenia, oraz wszelkiego rodzaju programy rezydujące, instalowane na stałe w pamięci operacyjnej (większość z nich zresztą także obsługuje przerwania procesora). Programy takie są często bardzo użyteczne, a w wielu przypadkach trudno się wręcz bez nich obyć. Aczkolwiek Asembler nadal pozwala tworzyć takie programy w sposób optymalny, minimalizując czas realizacji i zajmowaną pamięć, to jednak jego przewaga nie jest dzisiaj już tak miażdżąca.

Obok języka C także i popularny TURBO-Pascal w wersji 4.0 i 5.0 zapewnia bezpieczną obsługę praktycznie wszystkich przerwań, a tworzone w nim programy rezydujące często nie pożerają więcej niż 4 KB. Nie znaczy to bynajmniej, że zbędna staje się ogólna znajomość kultury mikroprocesora czy zasad korzystania z usług BIOS. Tym razem jednak akcent pada właśnie na słowo: ogólna. Szczegółami realizacyjnymi nie musimy się zajmować, zrobi to za nas kompilator TURBO-Pascala.

Obsługa przerwań asynchronicznych w TURBO-Pascalu jest możliwa dzięki temu, że większość podprogramów bibliotecznych TURBO-Pascala 4.0 i 5.0 jest współużywalna. Współużywalność oznacza, że dany podprogram może być w dowolnym punkcie przerwany i wywołany ponownie od początku np. przez procedurę obsługi przerwania, a następnie po zakończeniu obsługi przerwania, podjąć pracę i dokończyć przerwaną czynność bez żadnych zakłóceń. Współużywalne są m.in. wszystkie podprogramy arytmetyczne (chyba że korzystamy z koprocesora zmiennoprzecinkowego!) i operujące na łańcuchach. Można przyjąć za pewnik, że nie są współużywalne żadne mechanizmy, modyfikujące dane statystyczne, tzn. zlokalizowane w pamięci pod ustalonymi adresami lub korzystające z roboczych zmiennych statystycznych. Jest więc oczywiste, że nie mogą być współużywalne np. procedury zarządzające pamięcią dynamiczną.

Nie jest też gwarantowana współużywalność procedur wejścia-wyjścia i w ogóle programów korzystających z usług systemu operacyjnego. W jednozadaniowym z natury systemie MS-DOS/PC-DOS część procedur usługowych nie jest bowiem współużywalna.

Jeżeli program ma obsługiwać przerwania, musi zawierać przynajmniej dwa elementy: właściwy podprogram obsługi przerwania oraz program inicjujący, który za-

stępuje dotychczasowy wektor przerwania nowym wektorem, wskazującym na nasz podprogram obsługi. Do modyfikacji wektora można użyć procedury pascalowej **SetIntVec**.

Rozważmy przypadek najprostszy: program pascalowy organizuje obsługę przerwań na własny użytek. Nie trzeba obawiać się konfliktów z innymi programami, natomiast przed opuszczeniem programu należy bezwzględnie odtwarzać pierwotną wartość przyjętych wektorów przerwań (trzeba ją zapamiętać przed wpisaniem nowego wektora, np. używając funkcji **GetIntVec**). W przeciwnym razie ciągle aktywny podprogram obsługi przerwania zostanie znienacka zastąpiony przez inny kod, a następne przerwanie spowoduje awarię systemu. Awaria ta zapewne nie nastąpi od razu (mimo zakończenia programu pamięć operacyjna zachowuje swą zawartość), ale dopiero w chwili załadowania do zwolnionej pamięci następnego programu.

W drugim przypadku po inicjacji program jest pozostawiany w pamięci jako rezydujący i nie grozi mu zniszczenie przez programy ładowane do pamięci później. Odtwarzanie wektora przerwań jest też w ogólnym przypadku zbędne. Natomiast program rezydujący musi uwzględniać fakt współbieżnej pracy z innym oprogramowaniem, w szczególności — możliwość ewentualnego przepełnienia stosu procesora. Pisząc program w Asemblerze możemy operować stosem bardzo oszczędnie, ale kompilator TURBO-Pascala lokuje na stosie wszystkie zmienne robocze poza statycznymi (w oryginalnej nomenklaturze firmy Borland: *typed constant*).

Do deklarowania procedury obsługi przerwań służy dyrektywa: INTERRUPT, podawana bezpośrednio za nagłówkiem procedury. Nagłówek procedury musi mieć następującą, standardową postać (nazwa procedury może oczywiście być inna):

```
PROCEDURE Obsluga(Flagi,CS,IP,AX,BX,CX,DX,SI,DI,DS,ES,BP:word);
INTERRUPT;
BEGIN
    .........
END;
```

Wszystkie rejestry procesora są przekazywane jako pseudoparametry, tak że w procedurze można nie tylko odczytywać, ale i modyfikować ich zawartość (obie możliwości, a zwłaszcza druga, są oczywiście sensowne tylko w obsłudze przerwań programowych). W chwili wystąpienia przerwania i wywołania procedury obsługi wszystkie rejestry procesora zostaną automatycznie przechowane na stosie, a do rejestru DS wpisany adres segmentowy pascalowego segmentu danych. Dzięki temu procedura obsługi przerwania ma dostęp do wszystkich zmiennych globalnych (w tym do zmiennych statycznych). W chwili opuszczania procedury obsługi wszystkie rejestry zostaną oczywiście odtworzone, mówiąc żargonem: „zdjęte ze stosu".

Procedura obsługi przerwań może wywoływać inne procedury pascalowe, nie powinna jednak korzystać z żadnych standardowych funkcji wejścia/wyjścia, zwłaszcza odwołujących się do pamięci masowych, ani z jakichkolwiek funkcji systemu operacyjnego, związanych z obsługą urządzeń zewnętrznych. W ogóle odwołań do systemu operacyjnego lepiej unikać, a w każdym razie każde takie odwołanie warto przemyśleć. Nie wolno też korzystać z procedur pascalowych, zarządzających pamięcią dynamiczną. Wewnątrz procedury obsługi przerwania pozostaje wyłączona reakcja na dalsze przerwania (flaga obsługi przerwań IRQ procesora jest automatycznie kasowana w chwili wywołania programu obsługi przerwania), chyba że sami ją umożliwimy rozkazem maszynowym STI, np. użytym w instrukcji INLINE ($FB).

Napomknęliśmy o koprocesorze. W chwili wywołania procedury typu INTERRUPT nie są automatycznie zapamiętywane rejestry ewentualnego koprocesora. Tak więc próba wykonania w procedurze obsługi przerwania jakichkolwiek operacji z udziałem koprocesora, a więc działań na liczbach typu *single*, *double*, *extended*, *real* czy *comp*, może spowodować zakłócenie obliczeń zmiennoprzecinkowych w przerwanym programie. Zamierzając wykonywać operacje z udziałem koprocesora w procedurze obsługi przerwania należy więc bezwzględnie przechowywać na stosie kompletny status koprocesora. W praktyce powyższe rozważania można jednak na szczęście zaliczyć raczej do „gdybania", gdyż potrzeba operacji zmiennoprzecinkowych w procedurze obsługi przerwania należy do sytuacji raczej kuriozalnych.

Jakiego by tu użyć przykładu praktycznego? Niedawno majstrowaliśmy w Asemblerze zegar cyfrowy — dlaczego by dla porównania nie przećwiczyć tego problemu w TURBO-Pascalu? Poniższy program demonstruje praktyczne zastosowanie procedury obsługi przerwania do stworzenia zegara cyfrowego, podającego bieżący czas w prawym górnym rogu ekranu. Skorzystamy z faktu, że BIOS komputera periodycznie wywołuje przerwanie programowe nr $1C (dziesiętnie 28). Dzieje się to ok. 18 razy na sekundę, po każdym „tyknięciu" zegara systemowego. Przerwanie $1C nie jest „czystym" przerwaniem sprzętowym, tym niemniej jest ono wywoływane asynchronicznie z właściwej procedury obsługi sprzętowego przerwania zegarowego BIOS. Z naszego punktu widzenia jest to praktycznie równoważne przerwaniu sprzętowemu. Oto gotowy program. Jego podstawowym zadaniem jest „rozrzucanie" po ekranie losowych liczb, ale równolegle, co sekundę pulsuje w prawym górnym rogu „cyferblat" zegara:

```
USES CRT, DOS;
VAR wektor_pierw: pointer;
    Rejestry    : Registers;
    x           : integer;

    PROCEDURE Wyswietl_zegar;
    CONST Atrybuty_zeg= $7000;
          Adres_zegara: word=140;
          Adr_akt:      word=140;
          Seg_ekr:      word=$B800;  {dla karty HGC $B000}

    PROCEDURE Para_cyfr(x:byte);
    BEGIN
       MemW[Seg_ekr:Adr_Akt]:= 48+(x DIV 10)+Atrybuty_zeg;
       Adr_Akt:= Adr_Akt+2;
       MemW[Seg_ekr:Adr_Akt]:= 48+(x MOD 10)+Atrybuty_zeg;
       Adr_Akt:= Adr_Akt+2;
    END;

    PROCEDURE Dwukropek;
    BEGIN
       MemW[Seg_ekr:Adr_Akt]:= $3A+Atrybuty_zeg;
       Adr_Akt:= Adr_Akt+2;
    END;

    BEGIN
       Adr_akt:= Adres_zegara;
       Rejestry.AH:=$2C;
       MsDos(Rejestry);
       Para_cyfr(Rejestry.CH);   Dwukropek;
       Para_cyfr(Rejestry.CL);   Dwukropek;
       Para_cyfr(Rejestry.DH);
    END;
```

```
PROCEDURE Zegar_cyfrowy(Flagi, CS, IP, AX, BX, CX, DX,
                        SI, DI, DS, ES, BP: word);
INTERRUPT;
CONST licznik: word=0;
BEGIN licznik:= licznik+1;
      IF licznik>18
        THEN BEGIN licznik:= 0;
                   Sound(800); Delay(8); NoSound;
                   Wyswietl_zegar
             END
END;

BEGIN ClrScr;
      DirectVideo:= false;
      GetIntVec($1C, wektor_pierw);
      SetIntVec($1C, @Zegar_cyfrowy);
      REPEAT x:= Random(10000);
             GotoXY((x MOD 6)*12+1, (x MOD 17)+6);
             Write(x:10)
      UNTIL KeyPressed;
      SetIntVec($1C, wektor_pierw);
END.
```

Najpierw trzeba odczytać bieżący wektor przerwania i przechować go w zmiennej wskaźnikowej, a dopiero potem „przestawić" wektor przerwania na nową procedurę obsługi. Co więcej: przed zakończeniem programu trzeba koniecznie odtworzyć pierwotny stan wektora przerwania. W przeciwnym razie, mimo usunięcia programu pascalowego z pamięci, każde kolejne przerwanie będzie próbować wywołać procedurę obsługi pod dotychczasowym adresem, co doprowadzi do awarii systemu operacyjnego.

Odczyt czasu systemowego odbywa się w procedurze obsługi przerwania za pomocą funkcji usługowej DOS nr $2C, dostępnej za pośrednictwem przerwania $21. W tym kontekście jej użycie jest dozwolone, a niewątpliwą zaletą funkcji jest fakt, że wpisuje ona do odpowiednich rejestrów odpowiednio godziny, minuty itd.

Aby uniknąć korzystania ze standardowych mechanizmów wejścia/wyjścia została napisana procedura **Wyświetl-zegar**, wpisująca kody i atrybuty cyfr wprost do pamięci ekranu. Procedura ta odczytuje zarazem bieżący czas z systemu operacyjnego. Jest ona wywoływana ok. 1 raz na sekundę przez procedurę obsługi przerwania Zegar-cyfrowy. Procedura ta jest wywoływana ok. 18 razy na sekundę i zlicza te wywołania, korzystając ze statycznej zmiennej: **licznik**. Co 18 „tyknięć" na ekran jest wyprowadzany bieżący czas i wysyłany krótki sygnał dźwiękowy. Dla kart monochromatycznych należy zmienić parametr: **Seg__ekr** na $B000, odpowiednio do adresu segmentowego ich pamięci ekranu.

Nasz poprzedni program mógł działać tylko wewnątrz programu pascalowego. Przed opuszczeniem programu należało go koniecznie wyłączyć. Bardziej przydatny byłby zegar cyfrowy funkcjonujący dyskretnie, lecz stale i niezależnie od bieżącego programu pierwszoplanowego. Program taki musiałby więc być instalowany w pamięci operacyjnej na stałe. Umożliwia to procedura pascalowa **Keep**. Oprócz tego kończy realizację programu i podobnie jak **Halt** pozwala przekazać do systemu operacyjnego kod zakończenia.

Spróbujmy zaprogramować rezydujący zegar ekranowy w TURBO-Pascalu 4.0 lub 5.0. Jako dodatkowego usprawnienia żądamy możliwości określenia lokalizacji zegara na ekranie przy jego uruchamianiu jako pliku .EXE. Mogłoby się to odbywać przez podanie w linii zlecenia pary parametrów liczbowych, określających odpowiednio numer kolumny i wiersza ekranu, w którym ma znajdować się pole zegara.

Koncepcja samego zegara jest identyczna jak poprzednio. Ponieważ jednak nasz program ma działać równolegle z różnymi programami, niekoniecznie stworzonymi w TURBO-Pascalu, trzeba dążyć do usunięcia potencjalnych źródeł konfliktów. Przyjmiemy bezpieczną zasadę, że program nie będzie korzystać z żadnych funkcji usługowych BIOS i DOS. Jak jednak ustalić bieżący czas systemowy? Odczytamy go wprost z czterobajtowej komórki licznikowej, zlokalizowanej w pamięci operacyjnej pod adresem 40:6CH. Format licznika jest identyczny z formatem zmiennych typu longint, zatem najwygodniej będzie zadeklarować pod wskazanym adresem zmienną tego typu.

Licznik przedstawia liczbę cykli zegara systemowego, jakie upłynęły od godziny 0.00. Zegar systemowy „tyka" z częstotliwością 18,2065 Hz. Dzieląc licznik przez liczbę impulsów na godzinę (65 543), uzyskamy bieżącą godzinę. Dzieląc resztę z tego dzielenia przez liczbę cykli na minutę (1092) uzyskujemy liczbę minut, itd. Wprawdzie podane dzielniki nie są w rzeczywistości liczbami całkowitymi, ale wynikający z zaokrąglenia błąd jest zupełnie pomijalny. Rezygnując z operacji na liczbach rzeczywistych zmniejszamy zaś objętość programu typu .EXE oraz ograniczamy powodowane przez zegar straty czasu procesora. Tak jak poprzednio, program wpisuje co sekundę kody cyfr wprost do pamięci ekranu, ale adres segmentowy pamięci ekranu nie jest tym razem stałą, lecz zmienną typu word o nazwie **Seg__ekr**. Zrezygnowano też z akustycznej sygnalizacji upływu sekund:

```
{$M 1024,0,0}
PROGRAM Zegar_rezydujacy;
USES CRT, DOS;
CONST Atrybuty_zeg= $7000;
VAR Godziny, Minuty, Sekundy, Seg_ekr,
    Adres_zegara, Adr_akt: word;
    Lin, Kol, Stan         : integer;
    Czas                   : longint ABSOLUTE $40:$6C;
    Tryb                   : byte    ABSOLUTE $40:$49;

  PROCEDURE Wyswietl_zegar;

    PROCEDURE Para_cyfr(x:word);
    BEGIN
      MemW[Seg_ekr:Adr_Akt]:= 48+(x DIV 10)+Atrybuty_zeg;
      Adr_Akt:= Adr_Akt+2;
      MemW[Seg_ekr:Adr_Akt]:= 48+(x MOD 10)+Atrybuty_zeg;
      Adr_Akt:= Adr_Akt+2;
    END;

    PROCEDURE Dwukropek;
    BEGIN MemW[Seg_ekr:Adr_Akt]:= $3A+Atrybuty_zeg;
          Adr_Akt:= Adr_Akt+2;
    END;

    BEGIN
       Godziny:= Czas DIV 65543;
       Minuty := (Czas-Godziny*65543) DIV 1092;
       Sekundy:= (Czas-Godziny*65543-Minuty*1092) DIV 18;
       Adr_akt:= Adres_zegara;
       Para_cyfr(Godziny);     Dwukropek;
       Para_cyfr(Minuty);      Dwukropek;
       Para_cyfr(Sekundy);
    END;

  PROCEDURE Zegar_cyfrowy(Flagi, CS, IP, AX, BX, CX, DX,
                          SI, DI, DS, ES, BP: word);
  INTERRUPT;
  CONST licznik: word=0;
  BEGIN licznik:= licznik+1;
        IF licznik>18
          THEN BEGIN licznik:= 0; Wyswietl_zegar END;
  END;

BEGIN
  Val(ParamStr(1), Kol, Stan);
  IF (Stan<>0) OR NOT (Kol IN [1..73]) THEN Kol:= 73;
  Val(ParamStr(2), Lin, Stan);
  IF (Stan<>0) OR NOT (Lin IN [1..25]) THEN Lin:= 1;
  Adres_zegara:= (Lin-1)*160+(Kol-1)*2;
  IF Tryb=7 THEN Seg_ekr:=$B000 ELSE Seg_Ekr:=$B800;
  SetIntVec($1C, @Zegar_cyfrowy);
  Keep(0);
END.
```

Na większą uwagę zasługuje program główny, który pełni w naszym przypadku jedynie funkcję instalacyjną. Niech program po skompilowaniu jest zawarty w pliku ZEGARREZ.EXE. Po uruchomieniu program zakłada, że w linii zlecenia podano parę liczb, z których pierwsza (1..73) określa kolumnę ekranu, w której rozpoczyna się pole zegara, natomiast druga (1..25) — wiersz ekranu. Przykład:

C: ZEGARCYF 50 25

Ponieważ parametry wywołania są pobierane z systemu operacyjnego przez funkcję ParamStr jako tekst, procedura Val przetwarza je na liczbę. Gdyby podano tylko pierwszy parametr lub nie podano żadnego, wówczas funkcja ParamStr dostarczy zamiast nie istniejącego parametru łańcucha pustego, a procedura Val zasygnalizuje błąd. Spowoduje to przyjęcie standardowej wartości pominiętego parametru. Podobnie program zareaguje w razie podania wartości spoza dozwolonego przedziału.

Aby raz wygenerowany program działał poprawnie na komputerach wyposażonych w różne typy kart graficznych, program instalacyjny powinien badać typy kart i wybierać odpowiedni adres segmentowy pamięci ekranu. W praktyce wystarczy odczytać z BIOS tryb graficzny. Tryb nr 7 oznacza kartę monochromatyczną, inne tryby — karty grafiki barwnej. Chociaż „prawomyślna" metoda testowania trybu graficznego polega na wykorzystaniu przerwania BIOS nr 10H, tym razem odczytamy tryb pracy wprost z pamięci roboczej BIOS, gdzie jest zapisany w bajcie o adresie absolutnym 40H:49H. Potem można już przestawić wektor przerwania nr 1CH, ustawiając je na naszą własną procedurę obsługi. Zapamiętywanie pierwotnego wektora jest zbyteczne, gdyż nie zamierzamy go odtworzyć. Ostatnią czynnością jest opuszczenie programu z równoczesnym pozostawieniem go w pamięci jako rezydującego, do czego służy procedura **Keep**.

Na samym początku programu występuje dyrektywa rezerwacji pamięci operacyjnej ($M 1024,0,0). Oznacza ona, że program przy uruchomieniu zażąda tylko 1024 bajtów dla stosu (minimalna możliwa wartość) i ani bajta pamięci dynamicznej, co w programie obsługi przerwania jest sprawą oczywistą. Dyrektywa ta spełnia w programach rezydujących bardzo istotną rolę. Ponieważ procedura **Keep** zachowuje program w pamięci wraz z jego segmentem pamięci danych, stosem i pamięcią dynamiczną, to pamięć operacyjna stojąca do dyspozycji programów uruchamianych w dalszej objętości będzie pomniejszona o wszystkie te elementy. Należy więc dążyć do minimalizacji tych wartości. Jeśli program rezydujący nie korzysta ze zmiennych dynamicznych, można zarezerwować z pamięci dynamicznej. Jeżeli w programie rezydującym nie występują liczne i zagnieżdżone procedury z wieloma zmiennymi lokalnymi, to stos o pojemności 1024 bajty także okazuje się całkiem wystarczający.

Chcąc korzystać z wewnętrznego obszaru stosu programu trzeba pamiętać, że w chwili wywołania procedury obsługi przerwania do rejestru segmentu stosu SS nie jest automatycznie wpisywany adres segmentu stosu programu. W naszym przypadku było to zbędne, gdyż z uwagi na niewielkie zapotrzebowanie na stos korzystaliśmy ciągle ze stosu bieżącego, a obszar stosu programu pozostał nie wykorzystany. Gdyby zapotrzebowanie na stos było większe, skutki mogłyby być fatalne.

Gdyby w procedurze obsługi występowały zagnieżdżone procedury z parametrami lokalnymi, dla uniknięcia przepełnienia stosu systemowego należałoby korzystać ze stosu lokalnego (nie można liczyć na to, że w każdym przypadku stos bieżący zawiera dostateczny zapas pamięci). Jak to zrealizować? Przede wszystkim należy zadeklarować jako globalne cztery zmienne typu word, z których dwie posłużą do przechowania adresu segmentowego lokalnego stosu programu i pierwotnego stanu wskaźnika **SP**, a dwie następne — do przechowania stanu rejestrów **SS** i **SP** w momencie rozpoczęcia procedury typu **INTERRUPT**.

Program instalacyjny korzystając z funkcji **SSeg** i **SPtr** powinien przechować stan rejestrów **SS** i **SP** w programie głównym, gdyż łącznie wskazują one właśnie początek lokalnego stosu programu. Wykonując tę operację w programie głównym, a nie w procedurze uzyskujemy najlepsze wykorzystanie stosu, gdyż stos jest pusty i wskaźnik stosu **SP** wskazuje wtedy na sam jego początek. Na samym początku procedury typu **INTERRUPT** wstawiamy parę instrukcji z udziałem funkcji **SSeg** i **SPtr**, przechowując bieżący stan **SS** i **SP** oraz instrukcję **INLINE**, wpisującą do **SS** i **SP** adres lokalnego segmentu stosu. Na końcu procedury **INTERRUPT** wstawiamy następną instrukcję **INLINE**, odtwarzającą pierwotny stan stosu. Oto przykład adaptacji naszego programu do korzystania ze stosu lokalnego. Deklaracja zamiennych powinna wystąpić w programie głównym:

```
VAR Lokalny_SS, Buf_SS, Buf_SP: word;

  ..............................

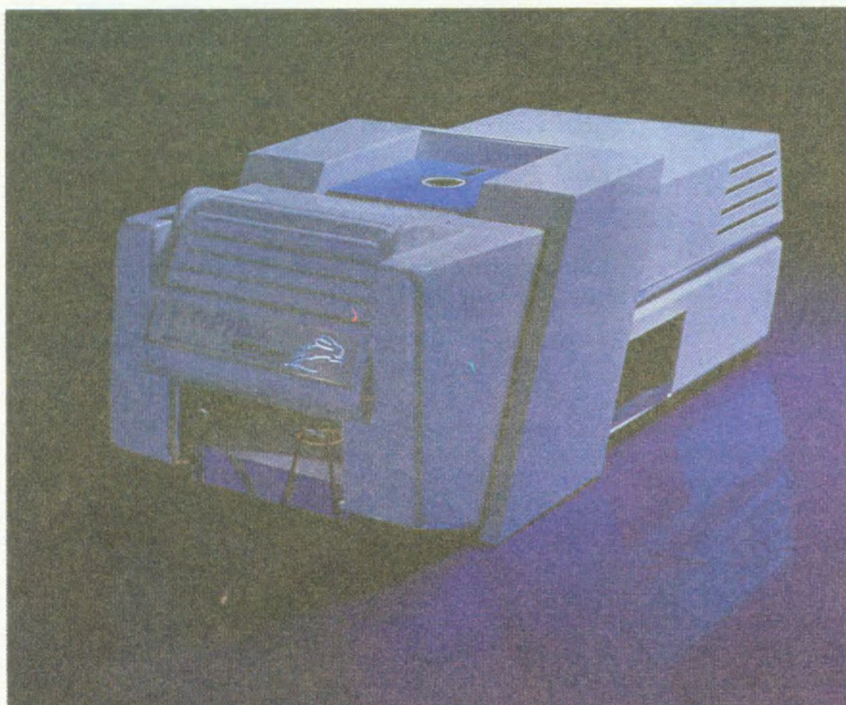
    PROCEDURE Zegar_cyfrowy(Flagi, CS, IP, AX, BX, CX, DX,
                            SI, DI, DS, ES, BP:word);
    INTERRUPT;
    CONST licznik: word=0;
    BEGIN Buf_SP:= SPtr; Buf_SS:= SSeg;
      INLINE($8E/$16/Lokalny_SS/     {MOV SS, Lokalny_SS}
             $8B/$26/Lokalny_SP      {MOV SP, Lokalny_SP} );
      licznik:= licznik+1;
      IF licznik>18
             THEN BEGIN licznik:= 0; Wyswietl_zegar END;
      INLINE($8E/$16/Buf_SS/         {MOV SS, Buf_SS}
             $8B/$26/Buf_SP          {MOV SP, Buf_SP} );
    END;

  .............................

  BEGIN
    Lokalny_SS:= SSeg; Lokalny_SP:= SPtr;
    Val(ParamStr(1), Kol, Stan);
  .............................
```

Teraz możemy pozwolić sobie na większą rozrzutność w gospodarce stosem.

Jeżeli w trakcie obsługi przerwania należy zezwolić na obsługę innych przerwań (może to być koniecznie np. przy korzystaniu z interfejsów RS-232C), to rozkaz **STI** należy trzeba by wstawić dopiero po „przestawieniu" rejestrów **SS** i **SP** na stos lokalny (po pierwszej instrukcji **INLINE**), natomiast przed odtwarzaniem stanu **SS** i **SP** należy wyłączyć przerwania rozkazem **CLI** (np. wstawiając na samym początku ostatniej instrukcji **INLINE** kod $FA). Trzeba bowiem pamiętać, że każde nowe przerwanie wymaga zapasu wolnej przestrzeni na stosie, a tą na pewno dysponujemy tylko na stosie lokalnym.



Fot. 1

ROLAND WACŁAWEK

TARGI CeBIT PO RAZ CZWARTY

Jeszcze 4 lata temu CeBIT był częścią targów w Hanowerze. Dziś stanowi samodzielną imprezę i zajmuje większość olbrzymich hal wystawowych na terenach targowych. Wasz wysłannik miał okazję obejrzeć tę imprezę dzięki gościnności organizatorów i uprzejmości Ambasady RFN. Imprezy tak potężne jak Targi CeBIT nie sposób nawet dokładnie zwiedzić w ciągu kilku dni, cóż dopiero opisać na kilku stronicach. Moja relacja będzie więc z natury rzeczy wybiórcza i subiektywna. Skoncentruję się głównie na „dolnych obszarach" prezentowanej oferty, obejmujących komputery osobiste i osprzęt do nich, i spróbuję przedstawić eksponaty, stanowiące obiecujące nowinki technologiczne, reprezentujące umacniające się tendencje, względnie potencjalnie interesujące dla polskiego użytkownika.

Tradycyjne komputery osobiste z procesorami 80286 i 80386 zgodne z AT i standardem PS-2 prezentowało co najmniej kilkadziesiąt firm, głównie azjatyckich, reprezentujących podobny poziom technologiczny. Wydaje się jednak, że ekspansja rodziny PS-2 postępuje wolniej, niż oczekiwano. Wynika to m.in. z faktu wprowadzenia przez koalicję firm alternatywnego standardu architektury EISA, zapewniającego walory zbliżone do mikrokanału IBM, ale nie odcinającego się od popularnej dziś architektury PC/AT, a więc pozwalającego dalej używać kart rozszerzeń i interfejsów itd. Promocja EISA była w Hanowerze dość silna, m.in. ze strony firmy COMPAQ. Standardem graficznym staje się jednak wyraźnie VGA. Firma Genoa oferuje m.in. kartę Super-VGA 5400 z pamięcią 512 KB i rozdzielczością do 1024 × 768 punktów przy 16 kolorach. Karta ma 16-bitową magistralę, ale może przełączać się automatycznie także w tryb 8-bitowy, co pozwala wstawić ją tak do PC/AT, jak i do XT. Popularne są obudowy typu pionowej wieży; wobec morderczej konkurencji wielką wagę przywiązuje się do estetyki i wzornictwa, nawet w przypadku urządzeń przeznaczonych zdecydowanie dla zaplecza, jak np. automatyczna kopiarka dyskietek marki CopyMaster z magazynkiem na 20 lub 40 dyskietek (fot. 1).

Trwa moda na komputery przenośne. Nikogo nie dziwi już bateryjny AT ze sztywnym dyskiem mieszczący się w teczce. Wielką sensacją był jednak przenośny komputer osobisty Sharp z procesorem 80386 (zegar 20 MHz; możliwe zainstalowanie koprocesora 80387) i płaskim monitorem barwnym LCD o przekątnej 14 cali (279 × 209 mm), fot. 2. Monitor ten zapewnia zgodność ze standardem VGA, tzn. rozdzielczość 640 × 480 punktów. Każda z 3 barw podstawowych może występować w 8 natężeniach, co w sumie daje 512 możliwych barw. Jakość obrazu jest urzekająca — w konfrontacji ekranów LCD z kineskopami te ostatnie utraciły ostatni atut, jakim była możliwość operowania pełną paletą kolorów. Oprócz rewelacyjnego ekranu komputer posiada świetną klawiaturę o 94 przyciskach, standardowo 2 MB pamięci RAM (można rozbudować do 8 MB), stację dysków 3.5 cala (1.44 MB) oraz dysk sztywny 40 MB z czasem dostępu poniżej 20 ms. Niełatwo znaleźć mikrokomputer stacjonarny o takich parametrach...

Firma ATARI poszła w swym komputerze przenośnym, a raczej kieszonkowym, w stronę maksymalnej miniaturyzacji. ATARI PC Folio waży wraz z bateriami tylko 450 gramów i ma rozmiary 180 × 90 × 25 mm, mieszcząc się w kieszeni marynarki (fot. 3). Mikroprocesor 80C88 (zegar 4.91 MHz) i pamięć operacyjna 138 KB (rozszerzalna do 640 KB) umożliwiają pracę pod kontrolą systemu MS-DOS 2.11. System ten wraz z obszernym oprogramowaniem uzyskowym (edytor, arkusz kalkulacyjny, terminarz, prosty bank danych) mieści się w pamięci ROM o pojemności 256 KB. „Zaszycie" systemu

w ROM jest konieczne, gdyż komputer nie ma stacji dyskietek, lecz tylko pamięć na kartach magnetycznych, przydatną raczej do archiwowania danych. Miniaturyzacja wymaga kompromisów. Klawiaturę o 63 przyciskach trudno uznać za komfortową (rozmiary!), ekran o 8 liniach po 40 znaków lub 240 × 64 punkty w grafice nie wystarczają do większych prac. ATARI nie zaniedbuje oczywiście także „dorosłych" wersji komputerów osobistych, jak PC-5 z procesorem 80386, nie mówiąc o stale rozbudowywanej rodzinie ST. Nowością jest transputerowa stacja robocza ATW, mogąca zawierać do 17 kart transputerowych i dysponująca w tej wersji niesamowitą mocą obliczeniową.

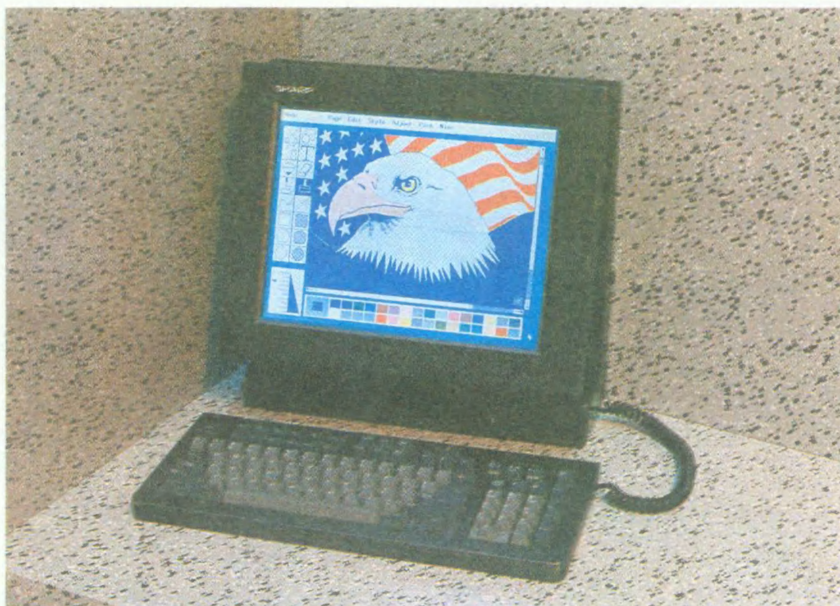
Rozwija się Apple Macintosh. Nowy model IIcx ma modularną architekturę, procesor Motorola 68030 z koprocesorem 68882 i pamięcią 2 MB RAM (z możliwością rozbudowy do 8 MB na płycie głównej) oraz całkiem nową obudowę (fot. 4). Pamięci masowe to stacja dyskietek 3.5 cala 1.44 MB i dysk sztywny 40 MB lub 80 MB.

Rodzina Commodore AMIGA także ma nowego członka. A2500UX to profesjonalny komputer z procesorem Motorola 68020 i koprocesorami 68881 i 68851, przeznaczony do pracy w systemie UNIX. Umożliwia to 32-bitowa magistrala i pamięć RAM 2 MB w wersji podstawowej. Jako pamięć masowa służy stacja dyskietek 3.5 cala, dysk 80 MB z czasem dostępu 19 ms i wbudowany na stałe streamer. Możliwości graficzno-muzyczne takie, jak w innych modelach Amigi (rozdzielczość od 320 × 200 do 640 × 512 punktów, do 4096 barw, 4-kanałowy syntetyzer dźwięku). Z komputerem jest dostarczany system operacyjny UNIX V system 3.1.

Dobrze przyjęły się małe, ręczne skanery. Protoplasta, tzn. popularny Handy-Skanner jest stale ulepszany. Pojawiły się wersje o szerokości skanowania 128 i 216 mm (np. HS-9100 firmy OADC), a także wariant analizujący obrazki barwne, na razie tylko o szerokości 64 mm. Nowe oprogramowanie pozwala analizować ilustracje o szerokości większej niż szerokość pola roboczego, umożliwiając dopasowanie kilku równoległych, oddzielnie skanowanych pasm. Pomysłem ręcznego skanera zaraziły się tak poważne firmy jak EPSON i Sharp, konstruując małe skanery barwne, nie wymagające jednak ręcznego „napędu". EPSON GT-1000 (fot. 5) analizuje obrazki o rozmiarach do 105 × 74 mm. Wystarczy położyć go na ilustracji (jest widoczna w przezroczystym okienku) i dalsza analiza odbywa się samoczynnie; odpada ręczne przesuwanie skanera. Rozdzielczość wynosi 200 punktów na cal i 256 poziomów szarości w trybie jednobarwnym. Skaner posiada elektroniczną „lupę", pozwalającą zmieniać skalę wybranego fragmentu rysunku w zakresie od 50 do 200 procent. Sharp JX-100 (fot. 6) to inny miniaturowy skaner, zdolny do analizy barwnych ilustracji o formacie A6 z rozdzielczością 200 punktów na cal. Także i tu wystarczy położyć skaner na ilustracji i wyzwolić proces analizy.

Głównym zastosowaniem obu skanerów będą zapewne tanie systemy DTP (Desktop Publishing). Technika DTP zaprezentowała się na targach w rozkwicie, o czym świadczyło m.in. liczne nowe oprogramowanie. Ostateczny efekt zależy tu jednak od drukarki. Jak na polskie warunki drukarki laserowe były dotąd drogie tak w zakupie, jak i w eksploatacji. Tymczasem w Hanowerze pojawiła się całkiem nowa rodzina drukarek pracujących na podobnej zasadzie jak laserowe, ale ... bez lasera! Technologię opracowała firma Casio, ale samą drukarkę prezentowała firma Qume (fot. 7). Za źródło światła służy lampa halogenowa, jego modulatorem jest specjalny element ciekłokrystaliczny. Całość uzupełnia samoogniskujący system optyczny. W ten sposób poza systemem przesuwu papieru z drukarki wyeliminowano wszelkie elementy ruchome, co obniża cenę i zwiększa żywotność i niezawodność urządzenia. Co więcej, bęben wraz z magazynkiem tonera nie stanowią już nierozłącznej całości, jak w wielu klasycznych drukarkach laserowych. Bęben i komorę cieplną można wymieniać oddzielnie, toner także uzupełnia się niezależnie, co pozwala kilkakrotnie obniżyć koszty eksploatacji (fot. 8). Drukarka jest mała i zwarta, lecz ma parametry odpowiadające typowym drukarkom laserowym: 6 stronic na minutę, 300 punktów na cal. Istnieje wiele wariantów, różniących się procesorem, w tym wariant zgodny ze standardem HP LaserJet+ i Series II oraz wersja z językiem PostScript. Drukarka ta, wkrótce zapewne dostępna i z innych źródeł, ma wszelkie szanse kosztować wkrótce poniżej 3000 DM, a jej walory zasługują na uwagę polskich klientów.

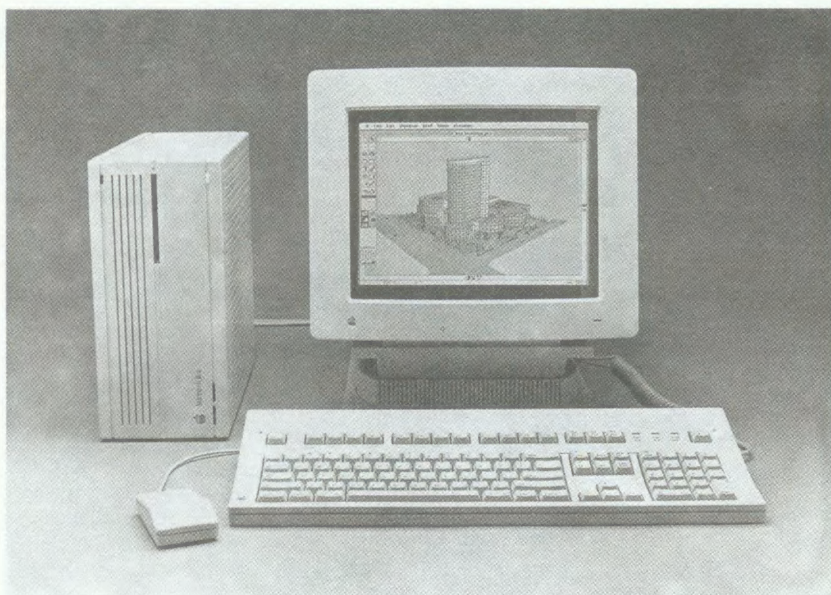
Alternatywą dla drukarki laserowej może być drukarka atramentowa EPSON



Fot. 2



Fot. 3



Fot. 4



Fot. 5

TSQ-4800 z głowicą o aż 48 niezależnie sterowanych dyszach. Urządzenie pracuje praktycznie bezszmerowo z prędkością 600 znaków lub 216 znaków/s w trybie najwyższej jakości. Rozdzielczość graficzna wynosi 360 × 360 punktów na cal, lista wbudowanych krojów pisma obejmuje 10 pozycji. Głowica jest automatycznie zabezpieczona przed wysychaniem, wskutek czego drukarka praktycznie nie wymaga bieżącej konserwacji.

Inną tendencję reprezentuje drukarka mozaikowa Seikosha SBP-10, przeznaczona do ciężkiej pracy i tak też wyglądająca (33 kg, fot. 9). Tylko 18 igieł, ale za to 800 znaków/s w trybie draft i 200 w trybie najwyższej jakości. Pamięć buforowa 64 KB. Zadrukowuje składankę lub luźne kartki pobierane z kasety, jak w drukarkach laserowych. Ręczną obsługę wspomaga wyświetlacz LCD i pole sterujące o 27 przyciskach. Popularna u nas firma Star pokazała 9-igłowe drukarki nowej serii FR-10/FR-15. Oto ich walory: prędkość druku do 25 znaków/s (draft) lub 63 znaków/s (NLQ), bufor o pojemności 31 KB, 8 krojów pisma, możliwość druku na pojedynczych kartkach bez potrzeby wyjmowania składanki z drukarki, kaseta o trwałości 5 mln znaków.

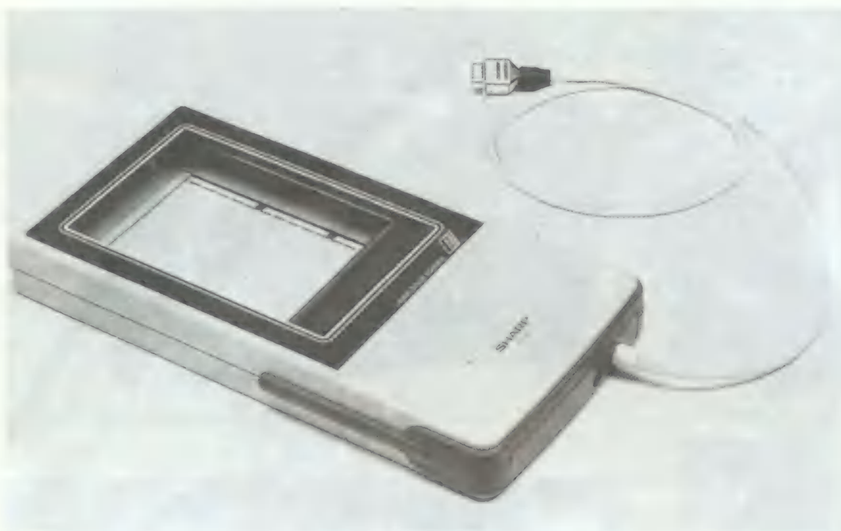
Wśród pamięci masowych dominują ciągle doskonalone dyski magnetyczne (np. dysk marki Fujitsu o średnicy 3 cale i pojemności 180 MB), ale rośnie udział obecności pamięci optycznych o wielkich pojemnościach. Oferowane są głównie dyski (CD) z jednorazowym zapisem (WORM), przydatne zwłaszcza do archiwowania danych i wypierające w tym zastosowaniu taśmę magnetyczną. Całą serię takich jednostek pamięci do różnych zastosowań, głównie z dyskami 5.25 cala o pojemności ok. 800 MB, przedstawiła firma Mitsui (fot. 10). Dysk wiruje w kasecie z prędkością 925 obrotów na minutę, średni czas dostępu wynosi 120 ms. Firma CBIS z USA przedstawiła specjalny serwer sieciowy z dyskami optycznymi (fot. 11), zawierający komputer klasy PC/AT oraz od 1 do 7 stacji dysków optycznych po 600 MB na jednostkę. Akceptowanych jest aż 28 różnych kart sieciowych. Dostarczane oprogramowanie realizuje funkcje NETBIOS i może współpracować z sieciowym oprogramowaniem Novell. Możliwe zastosowanie to m.in. udostępnianie w sieciach wielkich zbiorów archiwalnych i wszelkiego rodzaju katalogów, np. wyrobów lub firm, dostarczanych na dyskach optycznych przez wyspecjalizowane firmy marketingowe.

Coraz popularniejsze są wymienne dyski magnetyczne w kasetach systemu Bernoulli. Jednostka typu II/44 pozwala zmieścić na jednej kasecie do 44.5 MB i posiada inteligentny interfejs SCSI (fot. 12); dostępne są też jednostki o dwóch szczelinach dla wymiennych kaset dyskowych z łączną pojemnością 89 MB, przeznaczone m.in. dla systemów PC/XT/AT i Apple Macintosch. Średni czas dostępu wynosi tylko 32 ms, z zastosowaniem dodatkowej pamięci „podręcznej" (ang. *cache*) o pojemności 32 KB czas ten spada do 22 ms.

Pojawiają się tanie plotery do użytku w biurze i w domu. Prosty w konstrukcji i pozornie filigranowy ploter Fujitsu (fot. 13) o masie zaledwie 4,6 kg, ale może operować 6 kolorami na arkuszach papierowych lub foliowych różnego formatu włącznie z A3. Rozdzielczość całkiem niezła: 0,025 mm.

W Hanowerze wystawiono sporo osprzętu specjalistycznego, jak bezpośrednio sterowane przez program projektujący obwody drukowane frezarki do płytek prototypowych czy czytniki kart magnetycznych, np. kredytowych lub do kontroli ruchu załogi (fot. 14). Ciekawym rozwiązaniem był płaski pulpit-ekran LCD o rozdzielczości 640 × 400, sprzężony z digitizerem (firma Photron). Na jednym pulpicie można obserwować rysunek wyprowadzony przez komputer i odręcznie kreślić. Przykładowym zastosowaniem może być kontrola podpisów i ich porównywanie z wzorcem. Fujitsu oferowała skaner laserowy w nowej technologii opartej na oryginalnym układzie zwierciadeł. Skaner analizuje obraz na odległość, może więc z daleka odczytać np. kod prążkowy na obłym korpusie butelki (fot. 15), radzi też sobie świetnie z obiektami o małym kontraście. Dzięki niewielkiej promieniowanej przez laser energii urządzenie jest całkowicie bezpieczne.

Ciekawym drobiazgiem był miniaturowy automodem Worldport 1200 i 2400 firmy Touchbase Inc. Niewielkie pudełko można wstawić wprost w gniazdo interfejsu RS-232C na tylnej ściance komputera. Do połączenia z gniazdem telefonicznym służy złącze na innej ściance. Modem jest sterowany całkowicie programowo; może on samoczynnie wybierać numery i odpowiadać na wywołanie. Cechuje go duża odporność na błędy przesyłu. Wbudowany głośniczek pozwala na podsłuch transmisji. Kilka diod LED w obudowie syg-



Fot. 6



Fot. 7



Fot. 10



Fot. 8



Fot. 9



Fot. 11



Fot. 12

nalizuje różne stany pracy. Modem jest przystosowany do pracy w warunkach polowych, np. łącznie z komputerami przenośnymi. Do zasilania wystarcza mu wówczas pojedyncza bateryjka 9 V (fot. 16).

Interesującym uzupełnieniem starszych komputerów klasy PC/XT i AT może być karta sterownika dysków elastycznych MULTI Driver II firmy Techway, pozwalająca dołączyć dysk w każdym z popularnych formatów (5.25 cala 360 KB i 1.2 MB oraz 3.5 cala 750 KB i 1.44 MB). Do jednej



Fot. 13



Fot. 14



Fot. 15



Fot. 16



Fot. 17

karty można dołączyć do 4 stacji, każda z nich w dowolnie wybranym formacie. Ta sama firma oferowała też zewnętrzne stacje dysków 3.5 cala do PC/XT/AT (fot. 17).

Krótko o oprogramowaniu. W sprzęcie klasy PC/AT/PS2 rośnie popularność systemu Windows; m.in. Microsoft lansuje pracujący w nim rewelacyjny arkusz kalkulacyjny Exel. Duże zainteresowanie sieciami, przy czym wielu wytwórców orientuje się na system Novell. Wyczuwalnie rośnie zainteresowanie transputerami i w ogóle przetwarzaniem równoległym, chociaż ilościowo są to ciągle jeszcze procesy raczej marginalne. Rozkwit przeżywa system UNIX i jego mutacje, z tym że niewielu bawi się nim na typowych komputerach osobistych; UNIX króluje raczej na mini-komputerach, a zwłaszcza na wydajnych tzw. stacjach roboczych (ang. *workstation*). Graficznym standardem UNIX-a wyraźnie stał się pakiet X-Windows. Klasyczne, relacyjne bazy danych przestają już dziś wystarczać, więc wielu producentów oferuje systemy bez danych oparte na innych koncepcjach, jak np. sieciowo zorientowany system db-VISTA III firmy Raima Corp., operujący m.in. językiem dostępu do danych SQL i nie stosujący plików indeksowych, lecz tzw. wskaźniki zbiorów, opisujące odwzorowania rekordów typu „jeden do kilku". Daje to istotne przyspieszenie dostępu do danych zwłaszcza w systemach o złożonych schematach transakcji, obejmujących wiele oddzielnych plików.

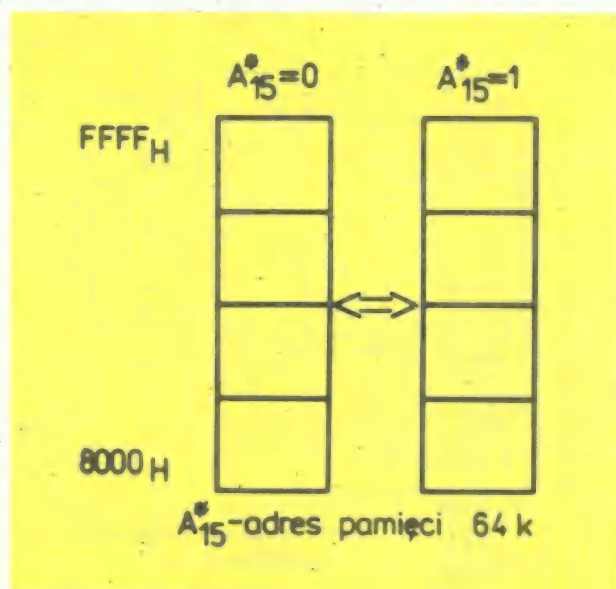


GRZEGORZ ZALOT

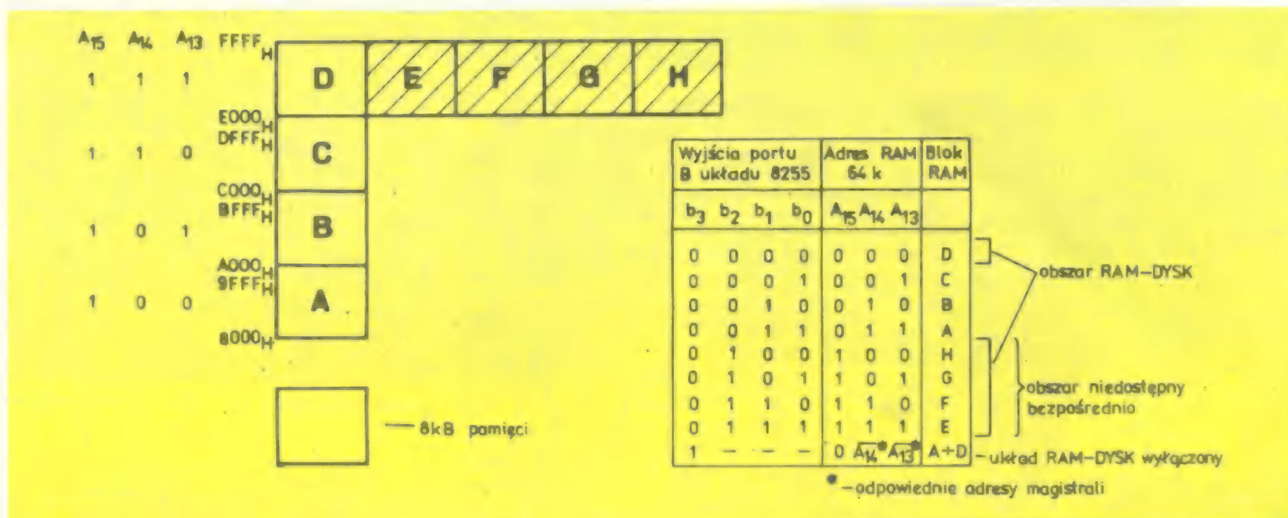
NAJPROSTSZY RAM-DYSK DLA ZX SPECTRUM

Wielu posiadaczy ZX Spectrum ma wbudowane w swoim komputerze elementy pamięci 64 KB zamiast typowo stosowanych elementów 32 KB. Dotyczy to szczególnie rozszerzonych modeli 16 KB, w których najczęściej montowano elementy 4164. W sytuacji takiej pozostaje nie wykorzystane 32 KB pamięci. Istnieje kilka możliwości „zagospodarowania" tej przestrzeni danych, czasem bardzo potrzebnej przy operowaniu na większych zbiorach informacji. Najprostszym wyjściem jest przełączanie najstarszego bitu adresowego — w ten sposób otrzymujemy dostęp alternatywnie do dwóch „połówek" pamięci, po 32 KB każda (rys. 1). Jest to niewątpliwie rozwiązanie najprostsze (opisywaliśmy je w numerze 3/86 „Młodego Technika"), ale fakt przełączania komórek pamięci począwszy od adresu 32768 wymaga, aby stos maszynowy (czyli na ogół również program w BASIC-u) mieścił się poniżej tego adresu. Tak drastyczne ograniczenie miejsca na program jest najczęściej nie do przyjęcia.

W poniższym artykule proponujemy inny sposób „zagospodarowania" dodatkowych 32 KB pamięci. Jest oczywiste, że dodatkowe komórki muszą być jakoś dostępne w normalnej przestrzeni adresowej procesora — konieczne jest więc wydzielenie pewnego specjalnego obszaru, za pomocą którego możemy się komunikować z dodatkowymi komórkami pamięci. Ze względu na prostotę konstrukcji obszar ten powinien być jak największy, a z uwagi na ograniczenie miejsca na program — jak najmniejszy. Pewnym kompromisem jest ustalenie wielkości tego obszaru na 8 KB — nie ogranicza to jeszcze zbytnio miejsca na program. Dodatkowe 32 KB pamięci będzie więc widziane jako 4 bloki po 8 KB przełączane tak, że będzie możliwy ich odczyt w wybranym obszarze przestrzeni adresowej procesora. Obszar ten najwygodniej umieścić na samej górze pamięci, czyli od adresu



Rys. 1. Jeden ze sposobów wykorzystania drugiej „połówki" pamięci RAM 64 KB



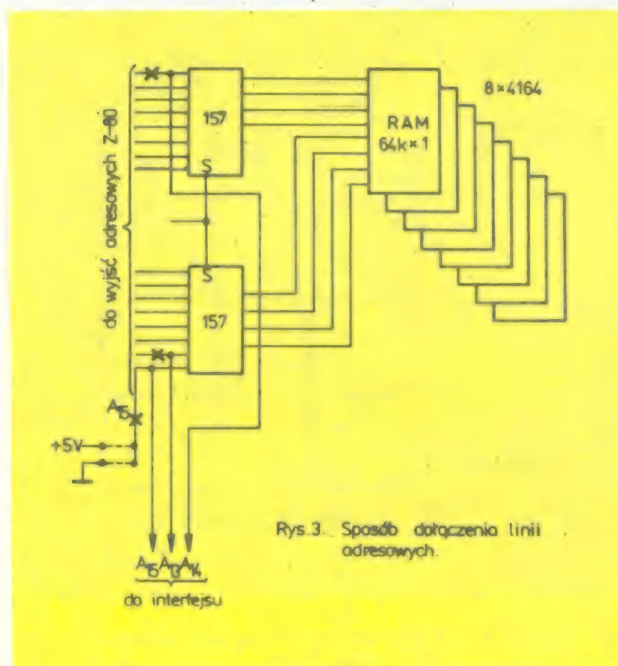
Wyjścia portu B układu 8255				Adres RAM 64 k			Blok RAM
b_3	b_2	b_1	b_0	A_{15}	A_{14}	A_{13}	
0	0	0	0	0	0	0	D
0	0	0	1	0	0	1	C
0	0	1	0	0	1	0	B
0	0	1	1	0	1	1	A
0	1	0	0	1	0	0	H
0	1	0	1	1	0	1	G
0	1	1	0	1	1	0	F
0	1	1	1	1	1	1	E
1	–	–	–	0	$\overline{A_{14}}$*	$\overline{A_{13}}$*	A÷D

Rys. 2. Sposób adresacji pamięci 64 K

E000H. Na program pozostaje więc ponad 30 KB pamięci, a dodatkowa pamięć ma obojętność 40 KB — jest to 5 bloków po 8 KB każdy.

Rysunek 2 przedstawia schematycznie organizację pamięci naszego komputera przy założonym sposobie dostępu do dodatkowych komórek. Obszar normalnie adresowany przez procesor, to bloki A, B, C i D. Natomiast bloki dodatkowe E, F, G i H mogą być umieszczane zamiennie z blokiem D w najwyższej części pamięci. Zauważmy, że blok D można łatwo adresować iloczynem adresów A14 i A13. W zasadzie powinniśmy jeszcze uwzględnić adres A15, ale układ sterowania dostępem do górnej połówki przestrzeni adresowej adres ten wykorzystuje i nie ma potrzeby dublowania odpowiedniej funkcji. A zatem po zdekodowaniu jedynek na liniach A14 i A13 należy na odpowiednie linie adresowe pamięci 64 KB podać wymagane dane tak, aby możliwe było odczytanie właściwych bloków. Zauważmy, że należy również odpowiednio wygenerować adres A15 dla pamięci 64 KB — w normalnym komputerze jest on stale równy 1 lub 0. A zatem konieczne jest zapisanie w rejestrze adresowym układu sterującego pamięcią trzech bitów, określających numer odczytywanego banku. W naszym układzie funkcję rejestru adresowego pełni element 8255 (port B). Pozostała część układu wykorzystana jest do zbudowania interfejsu równoległego dla drukarki.

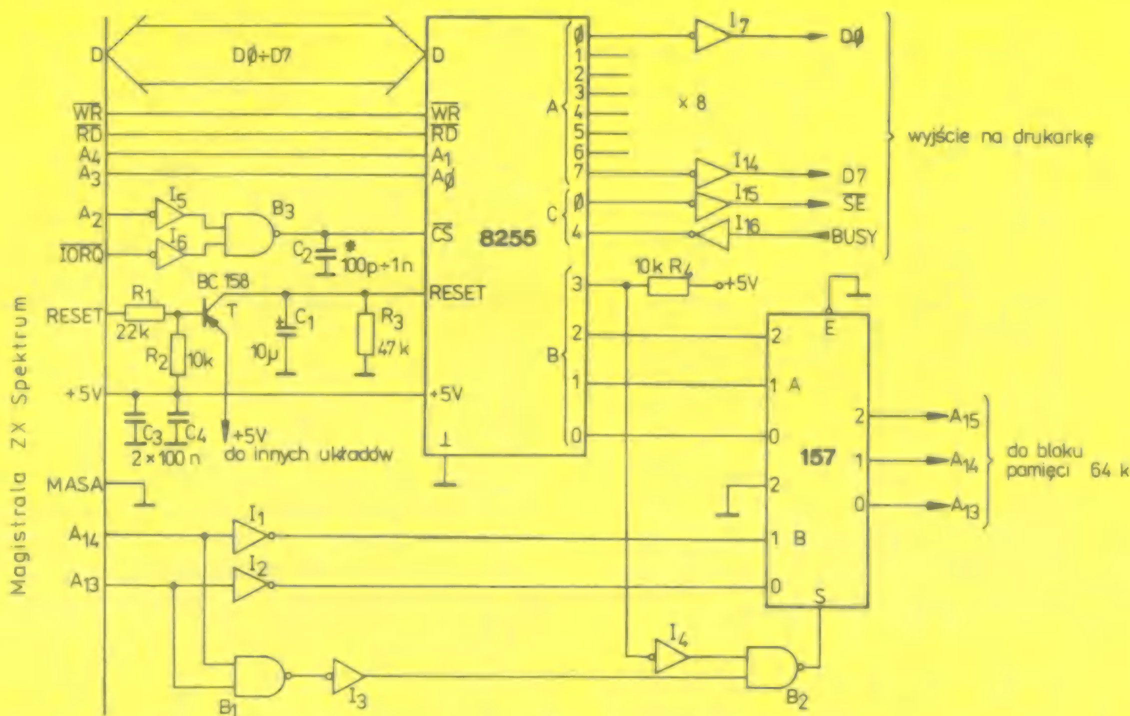
Rys. 3. Sposób dołączenia linii adresowych



Spójrzmy teraz na tabelkę na rys. 2. Trzy najmłodsze wyjścia portu B układu 8255 określają wspomniane trzy bity adresowe, natomiast bit b3 odblokowuje układ adresowania. Ta funkcja jest konieczna, gdyż po załączeniu zasilania wyjścia układu 8255 są zaprogramowane, jako wejścia i układy logiczne TTL odczytują stan ten jako logiczną jedynkę. Z kolei po zaprogramowaniu elementu na wyjściach pojawiają się zera — taka zmiana powoduje zawieszenie komputera czy jego wyzerowanie, jeżeli stos maszynowy umieszczony jest w bloku D (a jest tak zawsze po włączeniu komputera).

A zatem przy bicie b3 równym 0 bity b0 — b3 określają adres podawany na wejście pamięci 64 KB. Uwzględniając inwersję adresów A14 i A13, wprowadzaną przez układ sterujący, otrzymujemy dla poszczególnych kombinacji bitów b0—b3 odpowiednie bloki pamięci „widziane" przez procesor w miejscu bloku D. Można zatem wykorzystać nawet i te bloki, które znajdują się w normalnej przestrzeni adresowej — można teoretycznie zwiększyć w ten sposób pojemność RAM-DYSKU, lecz kosztem przestrzeni dostępnej dla programu w BASIC-u. W naszej wersji faktu tego nie wykorzystamy, przyjmując pojemność RAM-DYSKU równą 40 KB (czyli 40960 bajtów).

Konieczność wygenerowania odpowiednich adresów wymaga więc dotarcia do linii adresowych pamięci 64 KB. Jest to czynność stosunkowo prosta — rys. 3 przedstawia schematycznie sposób podłączenia się do pamięci. Podłączenie adresu A15 jest oczywiste — podany jest tam odpowiedni adres zamiast stałego sygnału 1 lub 0,



Rys. 4. Schemat ideowy interfejsu RAM-DYSKU i drukarki

co było konieczne przy pamięciach 32 KB odpowiedniej grupy. Natomiast adresy A14 i A13 dołączamy do odpowiednich wejść multiplekserów. Wejście adresu A14 dostępne jest na zworach ustalających adresowanie pamięci 32 KB, natomiast adres A13 na jednym z wyprowadzeń multiplekserów. Jeżeli układy multiplekserów w naszym komputerze są zamontowane na podstawkach, to wystarczy wyjęcie odpowiedniej i dolutowanie do niej przewodu. Pozostałe sygnały dostępne są na złączu krawędziowym.

Zlokalizowanie odpowiednich wyprowadzeń może jednak być trochę kłopotliwe — połączenia na druku są bardzo różne w zależności od wersji płytki. Najlepiej dokładnie prześledzić każde połączenie wizualnie i dodatkowo sprawdzić niskonapięciowym omomierzem (napięcie max 1,5V, prąd max 1mA — przy takich parametrach nie należy się obawiać uszkodzenia wrażliwych układów scalonych). Dla wersji komputera 4A i 4B rozkład wyprowadzeń pokazany jest na rys. 5. Zauważmy, że pewien niewielki kłopot może być jedynie z dotarciem do adresu A13 — pozostałe są dostępne w punktach lutowniczych zwor. Oczywiście zwory te należy usunąć. W przypadku adresu A13 wyjmujemy (lub wylutowujemy) odpowiednią nóżkę (zaznaczoną na rys. 5 dla wersji 4) i dolutowujemy przewód.

Pełny schemat ideowy układu sterowania pamięcią i interfejsu drukarki przedstawia rys. 4. Podstawowym elementem jest układ 8255. Pełni on funkcję dwóch 8-bitowych rejestrów wyjściowych, oraz dwóch 4-bitowych, z których jeden jest wejściowym, a drugi wyjściowym (port C). Przyłączenie do magistrali komputera jest standardowe — element jest wybrany, gdy linia A2 jest w stanie niskim. Zastosowano dekodowanie niepełne, dość często przyjmowane w ZX Spectrum. Linie adresowe A3 i A4 sterują wybieraniem odpowiednich rejestrów układu 8255. A zatem adresy poszczególnych rejestrów są następujące:

rejestr A — adres E3H
rejestr B — adres EBH
rejestr C — adres F3H
rejestr sterujący — adres FBH

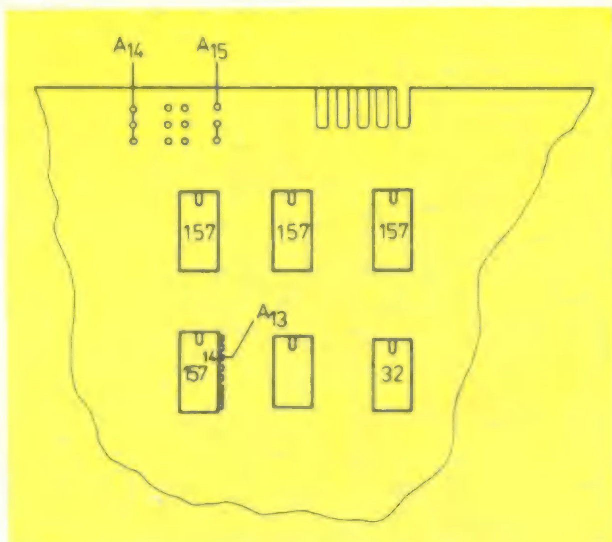
Poszczególne porty są wykorzystane w następujący sposób:

port A — rejestr wyjściowy interfejsu drukarki. Za nim włączone są dodatkowe inwertery dopasowujące wyjścia do wymagań przesyłu danych po kablu;
port B — rejestr wyjściowy układu sterowania pamięcią;
port C — młodsza połówka — układ wyjściowy do generowania strobu danych;
port C — starsza połówka — układ wejściowy do odczytu stanu zajętości drukarki.

Do rejestru sterującego należy po każdym wyłączeniu komputera wpisać odpowiednie słowo sterujące, uaktywniające właściwe funkcje rejestrów. W naszym przypadku ma ono wartość 88H. Jeżeli do rejestru tego nic nie wpiszemy, układ nie wpływa na pracę komputera, który zachowuje się, jak zwykłe ZX Spectrum z pamięcią 48 KB.

Właściwy układ generowania adresu dla pamięci 64 KB wykorzystuje multiplekser 74157 oraz bramki B1 i B2 wraz z odpowiednimi inwerterami. Działanie układu jest następujące (rys. 4):

Jeżeli układ nie jest zainicjowany, na wejściu inwertera I4 jest stan 1. Na jego wyjściu jest 0 i bramka B2 nie może być wysterowana. A zatem na wejście RAM 64 KB



Rys. 5. Fragment płytki w wersji 4A lub 4B z zaznaczonymi wyprowadzeniami adresów pamięci 64 KB

podane są zanegowane adresy A14 i A13 oraz 0 zamiast A15. Ta negacja adresów nie ma przy tym zupełnie żadnego znaczenia, gdyż nas interesuje jedynie jednoznaczność adresowania, a nie jego bezwzględna wartość. Jest ona poza tym konieczna dla zachowania adresów przy inicjowaniu układu 8255.

Po zainicjowaniu układu na wyjściu b3 pojawia się 0. Oznacza to podanie 1 na jedno wejście bramki B2. Jeżeli jednak adresy A14 i A13 nie będą równe 1, to bramka ta nie zostanie wysterowana i w adresacji pamięci nie zajdą żadne zmiany w stosunku do stanu sprzed inicjacji. Jeżeli jednak adresy A13 i A14 będą równe 1, czyli procesor odwoła się do interesującej nas części pamięci (powyżej 8000H oczywiście), to na linie adresowe pamięci 64 KB podany będzie stan wyjść b0—b2 portu B, czyli wybrany przez nas numer bloku pamięci. Oznacza to, że w takiej sytuacji możemy odwołać się do interesującego nas bloku drogą zapisania w porcie B odpowiedniego numeru bloku. Zauważmy przy tym, że możemy w ten sposób odwołać się do dowolnego bloku pamięci 64 KB (tabelka na rys. 2).

Układ interfejsu drukarki w zasadzie nie wymaga objaśnień, jest on poza tym praktycznie identyczny z opisanym w numerze 12/86 „Młodego Technika". Różnice dotyczą jedynie innych adresów rejestrów układu 8255. Można zatem łatwo wykorzystać oprogramowanie opisane w tymże numerze zarówno dla drukarki z wejściem CENTRONICS, jak i Logabax (DZM180).

Kilka słów na temat układu RESET. Element 8255 ma wejście RESET z aktywnym stanem wysokim, czyli odwrotnie niż procesor, trzeba zatem sygnał ten zanegować. To jednak nie wszystko — sygnał RESET elementu 8255 musi trwać nieco dłużej niż analogiczny sygnał procesora. Stąd właśnie kondensator 10 μF i rezystor 47 kΩ — przedłużają one sygnał ten o kilkaset milisekund.

Trzeba również zwrócić uwagę na oznaczony gwiazdką kondensator na wyjściu bramki B3. Otóż niektóre elementy 8255 (szczególnie produkcji radzieckiej), mają nieco inne zależności czasowe niż wymagane do współpracy z systemem Z-80. Konieczne jest wtedy lekkie opóźnienie narastającego zbocza sygnału /CS. Opóźnienie to jest zależne od egzemplarza układu i dobieramy je doświadczalnie w czasie uruchamiania całości. O tym jednak w dalszej części artykułu.

Układ sterownika pamięci zmontowany jest na dwustronnej płytce drukowanej. Można jednak, z uwagi na stosunkowo niewielką ilość połączeń po stronie elementów, zastosować również laminat jednostronny, a pozostałe połączenia, czyli głównie zasilanie, wykonać drutem w izolacji, lutowanym bezpośrednio do wyprowadzeń elementów. Schemat połączeń płytki przedstawia rys. 6.

Po starannym wlutowaniu elementów i sprawdzeniu, czy nie powstały ewentualne zwarcia, rozpoczynamy podłączanie układu. Najlepiej jest wykonać to na stałe, lutując do odpowiednich miejsc płytki cienkie przewody w izolacji. Wymaga to oczywiście odpowiedniego dopasowania obudowy. Można też wykorzystać złącze krawędziowe uzupełnione o niewielkie dodatkowe złącze sygnałów z multiplekserów pamięci 64 KB (lub też zmieniając przyporządkowanie niektórych praktycznie nie wykorzystywanych styków tego złącza). To już pozostawiamy majsterkowiczom — oczywiście nie takim zupełnie początkującym!!! Na początku nie podłączamy adresów A13 i A14, również nie przecinamy połączeń multiplekserów pamięci 64 KB — nie wylutowujemy zworek. Rozpoczynamy od sprawdzania pracy elementu 8255. W tym celu w BASIC-u zapisujemy słowo sterujące elementu rozkazem:

```
OUT 251, 136
```

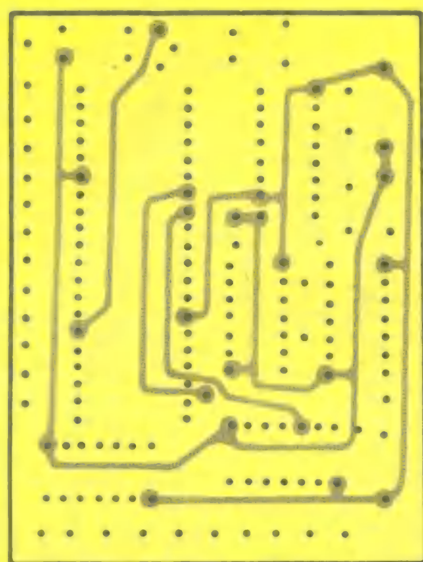
Teraz sprawdzamy, czy możliwe jest zapisanie w porcie A oraz B dowolnych danych. Posłużymy się znowu programem w BASIC-u:

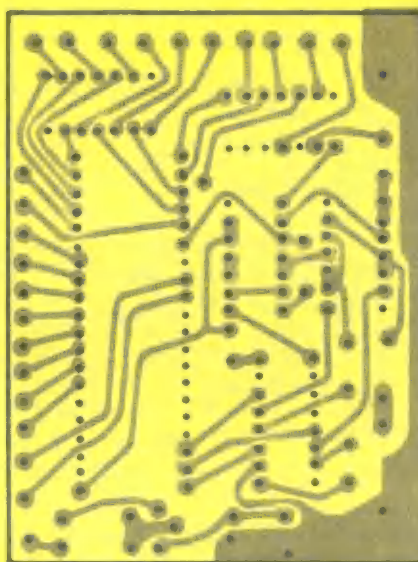
```
10 FOR N = 1 TO 255
20 OUT 227,N
30 NEXT N
40 GO TO 10
```

Adres 227 odpowiada rejestrowi portu A. Po uruchomieniu programu na poszczególnych liniach wyjściowych elementu (lepiej jest w tym przypadku sprawdzać sygnał za inwerterami) powinien pojawić się przebieg prostokątny o kolejno podwojonej częstotliwości — najwygodniej sprawdzić to za pomocą oscyloskopu, gdyż jednocześnie wykrywamy ewentualne zwarcia. Jest to znacznie trudniejsze za pomocą normalnej sondy TTL.

Następnie testujemy port B, zmieniając adres w instrukcji OUT na 235, identycznie jak poprzednio.

Port C testujemy w dwóch etapach. W pierwszym sprawdzamy linie wyjściowe, za pomocą opisanego programu, lecz z adresem 243. Oczywiście dotyczy to tylko 4 linii, a nawet tylko wyjścia inwertera I15. Natomiast linie wejściowe sprawdzamy testując w pętli stan portu o adresie 243 — przy zwarciu do masy wejścia inwertera I16 odczyt powinien rosnąć o 16. Inne dane mogą wskazywać na błędy w montażu, uszkodzone elementy lub też niewielkie niezgodności czasowe. Objawiają się one niewłaściwym zapisywaniem danych do rejestrów, jak również przeprogramowaniem elementu (przekłamania adresacji). W takim przypadku należy zwiększyć wartość pojemności przyłączonej do wyjścia bramki B3, przy czym górną granicą jest około 2 nF.





Rys. 6. Widok płytki drukowanej układu sterowania pamięcią i interfejsu drukarki

Teraz można już wykonać połączenia linii adresowych procesora i bloku 64 KB. Tu jest wymagana szczególna dokładność, połączenia należy starannie sprawdzić. Następnie uruchamiamy komputer i sprawdzamy, czy dostępna jest cała pamięć instrukcją:

CLEAR 65535

Nie powinno być błędu. Następnie inicjujemy element 8255 rozkazem:

OUT 251, 136

Teraz zapisując do portu 235 kolejno liczby 0, 4, 5, 6, i 7 aktywizujemy poszczególne 5 bloków naszego RAM-DYSKU. Proste operacje zapisu i odczytu BASIC-a pozwalają sprawdzić, czy rzeczywiście nie ma przekłamań i czy rzeczywiście pod tym samym adresem w poszczególnych blokach można przechowywać różne dane.

To już koniec etapu uruchamiania części sprzętowej RAM-DYSKU. Można teraz wykorzystać prosty program napisany w asemblerze, umożliwiający szybkie przerzucanie bloków danych (bajtów) z pamięci leżącej w obszarze poniżej E000H do przestrzeni RAM-DYSKU i odwrotnie, przy czym danymi będzie adres początku obszaru w RAM (bezwzględnie), adres względny w RAM-DYSKU (liczony od 0 do 40959), długość bloku oraz kierunek transmisji. Sama transmisja realizowana jest bardzo szybko, znacznie szybciej, niż trwa zapisanie odpowiednich danych.

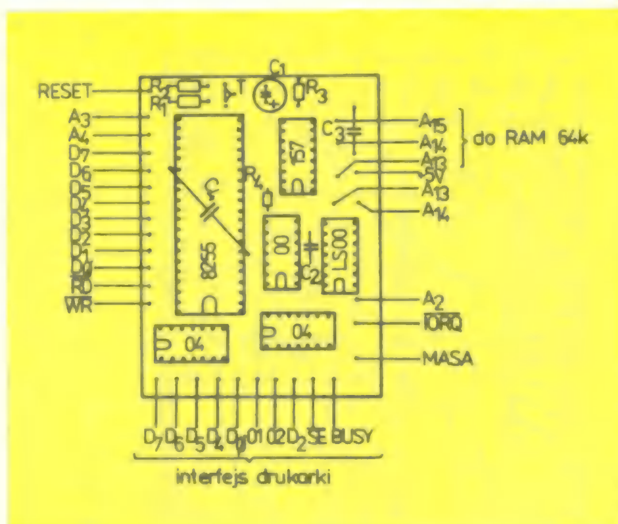
Postać źródłową tego programu przedstawia listing 1. Na początku umieszczono słowa, w których zapisać należy parametry transmisji. Są to, po kolei: adres początku obszaru pamięci komputera, adres początku obszaru w RAM-DYSKU, długość bloku danych oraz kierunek transmisji (0 oznacza odczyt z dysku, czyli zapis do pamięci komputera, a 1 zapis do RAM-DYSKU).

Początek programu oznaczony jest etykietą START. Najpierw sprawdzana jest poprawność danych przez podprogram ERRDAN — linie od 630. Dane są uważane za poprawne, jeśli długość przepisywanego obszaru jest różna od zera (zero dla instrukcji LDIR oznacza 65536!!), jeżeli obszar pamięci komputera nie pokrywa się z obszarem RAM-DYSKU oraz jeżeli nie jest przekroczona pojemność RAM-DYSKU. Należy zatem sprawdzić, czy końcowy adres obszaru RAM komputera jest mniejszy od adresu początku obszaru, w którym odczytujemy kolejne bloki dodatkowej pamięci (E000H). Sprawdzamy to dodając do adresu początkowego długość obszaru — linia 680. Brak przeniesienia (co oznaczałoby przekroczenie wartości FFFFH) oraz wynik mniejszy od E000H oznaczają poprawność danych. W przeciwnym wypadku następuje skok do etykiety BLAD, powodującej zatrzymanie realizacji programu z komunikatem „PARAMETER ERROR".

Podobnie sprawdzana jest poprawność danych dla RAM-DYSKU, różnica dotyczy tylko innej wartości maksymalnego adresu, w tym przypadku A000H (40960 — pojemność naszego RAM-DYSKU).

Po sprawdzeniu poprawności danych następuje inicjacja elementu 8255 — linie 100, 110. Linie poszczególnych portów zostają w tym momencie odpowiednio zaprogramowane.

Od etykiety OPER rozpoczyna się właściwy program transmisji danych. Na początku sprawdzane jest, czy cały obszar danych zawiera się w jednym bloku RAM-DYSKU. W tym celu obliczamy numer banku dla adresu początkowego — linia 130, 140. Procedura OBNRBA wyznacza odpowiedni numer banku w zależności od adresu przekazanego w parze HL. Numer ten jest jednoznacznie określony przez trzy najstarsze bity tego adresu, czyli rejestru H. Po przepisaniu go do akumulatora trzykrotnie rotujemy tak, aby interesujące nas bity były na najmłodszych pozycjach. Bity te jednak w interesującym nas przedziale adresów odpowiadają liczbom 0—4, natomiast do portu B elementu 8255 należy zapisać



Rys. 6a. Schemat montażowy płytki drukowanej — widok od strony elementów

odpowiednio 0, 7; 6, 5, 4. Konieczna operacja, to arytmetyczne zanegowanie zawartości akumulatora, a następnie zamaskowanie starszych bitów. Wynik zwracany jest w akumulatorze.

Aktualny numer banku zapisywany jest teraz do portu B (linia 150) oraz zapamiętany na stosie. Teraz sprawdzamy, w jakim banku zawiera się koniec obszaru RAM-DYSKU. Adres końcowej komórki obliczamy dodając do adresu początkowego długość (linia 180) oraz odejmując 1 (linie 190, 200). Ponownie wywołujemy procedurę OBNRBA i porównujemy numer banku z obliczonym poprzednio (linia 230). Jeżeli numer jest ten sam, skaczemy bezpośrednio do procedury przepisywania obszarów pamięci DUMP.

Procedura DUMP (linia 830) rozpoczyna się od wyznaczenia rzeczywistego adresu początku interesującego nas obszaru RAM-DYSKU w przestrzeni adresowej procesora. Zauważmy, że wystarczy w tym celu trzy najstarsze bity tego adresu ustawić na 1 — tę czynność wykonuje fragment do linii 860.

Operacje następne, to załadowanie do odpowiednich rejestrów danych dla instrukcji LDIR. W parze HL mamy już adres w odpowiednim bloku RAM-DYSK, do DE ładujemy adres początku bloku pamięci, do BC długość. Należy jeszcze określić kierunek transmisji — w tym celu testujemy komórkę KIER. Jeżeli jest zero, realizujemy odczyt. Przy zapisie należy jedynie zamienić między sobą pary DE i HL.

Jeżeli jednak okaże się, że koniec obszaru w RAM-DYSKU leży w innym bloku, konieczne jest ładowanie danych w kilku (minimum dwóch) etapach. W pierwszym ładowane są dane aż do końca bieżącego bloku. Odpowiednie obliczenia realizuje fragment od linii 280 do 410.

Najpierw obliczany jest adres pierwszej komórki następnego banku. W tym celu tworzony jest adres, w którym bity od zerowego do 12 włącznie są zerami, a trzy najstarsze powstają z odpowiednich trzech bitów aktualnego adresu, po dodaniu poprawki odpowiadającej naj-

```
 10 ;PROGRAM OBSLUGI RAM-DYSKU 40 kB
 20               ORG  23296
 30 ADRRAM        DEFW 0
 40 ADDISC        DEFW 0
 50 DLUG          DEFW 0
 60 KIER          DEFB 0 ;0-ODCZYT Z DYSKU
 70 DLUG1         DEFW 0
 80
 90 START         CALL ERRDAN
100               LD   A,#88
110               OUT  (#FB),A
120
130 OPER          LD   HL,(ADDISC)
140               CALL OBNRBA
150               OUT  (#EB),A
160               PUSH AF
170               LD   BC,(DLUG)
180               ADD  HL,BC
190               LD   BC,-1
200               ADD  HL,BC
210               CALL OBNRBA
220               POP  BC
230               CP   B
240               JR   NZ,ZMBANK
250               CALL DUMP
260               RET
270
280 ZMBANK        LD   L,0
290               LD   A,(ADDISC+1)
300               AND  #E0
310               ADD  A,#20
320               LD   H,A
330               LD   DE,(ADDISC)
340               AND  A
350               SBC  HL,DE
360               LD   DE,(DLUG)
370               LD   (DLUG),HL
380               EX   DE,HL
390               AND  A
400               SBC  HL,DE
410               LD   (DLUG1),HL
420               CALL DUMP
430               LD   HL,(DLUG1)
440               LD   DE,(DLUG)
450               LD   (DLUG),HL
460               LD   HL,(ADRRAM)
470               ADD  HL,DE
480               LD   (ADRRAM),HL
490               LD   HL,(ADDISC)
500               ADD  HL,DE
510               LD   (ADDISC),HL
520               JR   OPER
530
540 ;OBLICZENIE NUMERU BANKU
550 OBNRBA        LD   A,H
560               RLCA
570               RLCA
580               RLCA
590               NEG
600               AND  #07 ;NUMER W A
610               RET
620
630 ERRDAN        LD   HL,(ADRRAM)
640               LD   BC,(DLUG)
650               LD   A,C
660               OR   B
670               JR   Z,BLAD
680               ADD  HL,BC
690               JR   C,BLAD
700               LD   A,H
710               CP   #E0
720               JR   NC,BLAD
730               LD   HL,(ADDISC)
740               ADD  HL,BC
750               JR   C,BLAD
760               LD   A,H
770               CP   #A0
```

```
780                RET  C
790 BLAD           RST  8
800                DEFB 25 ;PARAMETER ERROR
810
820 ;PRZEPISYWANIE BLOKU
830 DUMP           LD   HL,(ADDISC)
840                LD   A,H
850                OR   #E0
860                LD   H,A
870                LD   DE,(ADRRAM)
880                LD   BC,(DLUG)
890                LD   A,(KIER)
900                CP   0
910                JR   Z,ODCZYT
920                EX   DE,HL
930 ODCZYT         LDIR
940                RET
950 ;KONIEC
```

młodszemu z tych bitów. Jest to zatem praktycznie inkrementacja liczby, jaką tworzą trzy najstarsze bity adresu. Po odjęciu od tego adresu wartości adresu początku bloku danych w RAM-DYSKU otrzymujemy ilość komórek od tego miejsca do końca bloku. Jest to zatem długość bloku danych, jaki można przesłać bezpośrednio przy pomocy instrukcji LDIR. Tę wartość zapisujemy do słowa DLUG, a do pomocniczego słowa DLUG1 wpisujemy ilość bajtów, które jeszcze pozostaną do przepisania. Teraz można już bezpośrednio wywołać procedurę przepisywania, z nową wartością długości.

Po przepisaniu części danych należy odtworzyć nowe wartości początkowe adresu w pamięci i RAM-DYSKU oraz długość pozostałego bloku danych. W naszym programie po kolei wpisujemy aktualną wartość długości (linia 450), obliczamy nowy adres w pamięci (linia 478, 480) i w RAM-DYSKU (linia 500, 510). Mamy zatem odtworzone dane wyjściowe i skaczemy do początku procedury tak, jakbyśmy rozpoczynali transmisję od początku. Program zatem jeszcze raz sprawdzi konieczność zmiany bloków i odpowiednio podejmie decyzję. Ten fragment programu można nieco skrócić, wykorzystując zmienione dane rejestrów po instrukcji LDIR, uwzględniając konieczną w przyjętym algorytmie korekcję rejestru H. Nie zrobiliśmy tego z uwagi na przejrzystość programu — poprawkę tę polecamy bardziej zaawansowanym w programowaniu w asemblerze.

Program zatem powtórnie analizuje wartości wyjściowe od początku. Proces ten kończy się wtedy, gdy ostatni blok danych zawiera się w jednym bloku RAM-DYSKU.

Program obsługi RAM-DYSKU nie jest relokowalny — w naszym przypadku umieściliśmy go w buforze drukarki (dyrektywa ORG 23296) — zajmuje on niespełna 200 bajtów. Oczywiście, w razie potrzeby można go ulokować w innym miejscu pamięci, oczywiście nie w obszarze najwyższych 8 KB, gdzie mamy obszar komunikacji z RAM-DYSKIEM. Ten problem pozostawiamy jednak użytkownikom.

Jak wykorzystać nasz program? Najpierw należy w odpowiednich komórkach zapisać dane wyjściowe. Zakładając, że program jest umieszczony od adresu A, mamy następujące adresy komórek roboczych:

A, A+1 — adres początku obszaru w przestrzeni pamięciowej komputera,

A+2, A+3 — adres w obszarze RAM-DYSKU, liczony względnie, czyli od zera,

A+4, A+5 — długość przepisywanego obszaru danych,

A+6 — kierunek transmisji: 0 oznacza odczyt z RAM-DYSKU, każda inna wartość — zapis,

A+9 — adres startu programu.

W przypadku adresów oraz długości mamy zarezerwowane każdorazowo dwa bajty (liczby większe od 255). Ponieważ jednak spod BASIC-a nie możemy jednocześnie zapisać dwóch bajtów, należy dokonać odpowiednich obliczeń. Przykładowy podprogram zapisu liczby D w komórkach AD i AD+1 jest następujący:

```
10 POKE AD,D-256*INT (D/256)
20 POKE AD+1,INT (D/256)
30 RETURN
```

Po zapisaniu danych uruchamiamy transmisję przez: RANDOMIZE USR (A+9)

Należy zauważyć, że przed każdym następnym wywołaniem procedury konieczne jest odtworzenie wartości adresów oraz długości bloku — w przeciwnym wypadku możliwe jest powstanie dużych błędów, wynikających z faktu, że program przy zmianie banków zmienia wartości komórek roboczych.

Jeżeli w komórkach roboczych zapiszemy błędne dane (czyli np. zerową długość lub próbę zapisu do pamięci pod adresem przekraczającym granicę najstarszych 8 KB, czy też przekroczenie pojemności RAM-DYSKU), to wykonywanie programu zostanie przerwane z komunikatem „PARAMETER ERROR".

Opisane układy sterowania pamięcią i interfejsu mogą współpracować z programami obsługi drukarki z interfejsem równoległym opisanym w n-rze 12/86 „Młodego Technika". Należy jedynie zmienić adresy rejestrów układu 8255, zgodnie z podanymi wcześniej.

Wykaz elementów:

1. Układy scalone:
— Intel 8255 lub odpowiedniki
— 74LS00 — 2 szt.
— 74LS04 — 2 szt.
— 74157 — 1 szt.

UWAGA!
Można bez większego ryzyka zastosować w opisanym układzie elementy serii standardowej 74 (czyli 7400 i 7404), za wyjątkiem elementu 74LS00 zaznaczonego na rys. 6a. Jest to konieczne, gdyż element ten ma po dwa wejścia podłączone do linii adresowych (bramka B1 oraz inwertery I1, I2 oraz I5). Pozostałe elementy nie wnoszą obciążeń, które mogłyby spowodować uszkodzenie czy niewłaściwą pracę komputera, oczywiście przy założeniu, że nie dołączamy jednocześnie innych interfejsów, dodatkowo obciążających linie procesora.

2. Tranzystor T — BC158 lub odpowiednik
3. Rezystory:
R1 — 22k, R2 — 10k, R3 — 47k, R4 — 10k
dowolnej mocy, najlepiej jak najmniejsze
4. Kondensatory:
C1 — 10μF/16V elektrolityczny
C2 — dobierany, 100pF ÷ 2.2nF
C3, C4 — 100 nF MKSE, przy czym jeden z nich lutujemy bezpośrednio do doprowadzeń zasilania elementu 8255.

C-64 JAKO MIERNIK CZĘSTOTLIWOŚCI

WOJCIECH ŻUREK

Uzupełniając komputer Commodore C-64 o opisany układ zewnętrzny (interface) i prosty program, możemy uczynić z niego miernik częstotliwości w zakresie od 1Hz do około 10 MHz. Miernik ten w zależności od wariantu wykonania mierzy sygnał TTL lub dowolny sygnał zmienny o amplitudzie większej od 100 mV. Posiada automatyczną zmianę zakresu pomiarowego, a cena elementów potrzebnych do jego wykonania nie przekracza 3000 zł.

Rozważania konstrukcyjne

Pomiar częstotliwości polega na zliczaniu ilości okresów przebiegu w określonej jednostce czasu. W naszym przypadku impulsy zliczane są w komputerze przez okres jednej sekundy. Czas zliczania, tzw. czas bramkowania, generowany jest przez wewnętrzny zegar komputera. Natomiast licznik impulsów zrealizowany jest programowo. Mimo że program licznika napisany jest w języku maszynowym, to dla wyższych częstotliwości czas jego wykonywania może być porównywalny, a nawet większy od czasu trwania okresu przebiegu wejściowego. Ogranicza to częstotliwość sygnału wejściowego do około 10 kHz (10 000 impulsów w ciągu sekundy). Częstotliwości większe musimy podzielić przed podaniem do komputera. Zakładając maksymalną częstotliwość mierzonych przebiegów jako 10 MHz musimy zastosować podział przez 1000. Z drugiej strony im większa liczba zliczonych impulsów tym większa dokładność pomiaru. Wygodnie jest więc dysponować sygnałami o częstotliwościach f, f/10, f/100, f/1000 i tak je podawać na wejście licznika, by zliczyć jak największą liczbę impulsów. Jeżeli oprócz sygnałów cyfrowych chcemy mierzyć sygnały analogowe to dzielnik częstotliwości musimy uzupełnić o odpowiedni wzmacniacz. Powinien to być wzmacniacz szerokopasmowy zapewniający odpowiednie wzmocnienie i dostosowanie sygnałów analogowych do standardu TTL. Przystawkę najlepiej podłączyć do komputera poprzez USER PORT podając sygnał o częstotliwości f na wejście PB7, f/10 na PB6, f/100 na PB5, f/1000 na PB4 i zasilić ją napięciem +5 V z tego portu. Przy takim zasilaniu należy pamiętać, że producent ustala maksymalny prąd tego wyjścia na 100 mA. Dokładność miernika jest lepsza niż 1% przy spełnieniu warunku, że licznik nie zliczy więcej niż 1000 impulsów. Uwaga ta nie jest słuszna dla najniższego zakresu; w tym bowiem przypadku układ nie ma już możliwości zmiany stopnia podziału przebiegu wejściowego w celu zwiększenia dokładności.

Opis konstrukcji

Schemat ideowy przedstawiono na rys. 1. Mierzony sygnał podawany jest przez opornik i kondensator na wejście dwustopniowego wzmacniacza tranzystorowego T1/T2. Kondensator wejściowy utworzony jest z dwu elementów C1/C2, jeden to kondensator elektrolityczny, a drugi ceramiczny. Takie połączenie zapewnia bowiem przenoszenie szerokiego pasma sygnałów (indukcyjności własne kondensatorów elektrolitycznych znacznie tłumią sygnał wcz.). Regulowany opornik R4 wraz z opornością wejściową wzmacniacza (około 10 kΩ) tworzy dzielnik napięcia regulujący wzmocnienie układu. Dla sygnałów stałoprądowych wzmacniacz objęty jest silnym ujemnym sprzężeniem zwrotnym R7 i R12, ustalającym jego punkt pracy. By opornik R12 nie zmniejszał wzmocnienia sygnału mierzonego zbocznikowano go pojemnością C4/C5. Z wyjścia wzmacniacza sygnał podawany jest przez wtórnik emiterowy T3 na wejście przerzutnika Schmitta. Wtórnik zapewnia dopasowanie wysokiej oporności wyjściowej wzmacniacza do niskiej oporności wejściowej układów TTL, natomiast przerzutnik zapobiega generowaniu paczek impulsów przy przebiegach wolnozmiennych. Dioda D1 ułatwia blokowanie tranzystora T3 i przez to wymuszenie odpowiedniego poziomu napięcia zera logicznego na wejściu TTL. Dodatkowe polepszenie parametrów wzmacniacza uzyskano stosując płytkie dodatnie sprzężenie zwrotne R10 i R11 (oporność R11 wynosi 4,7 Ω). Sygnał wyjściowy przerzutnika o częstotliwości fx podany jest na wejście PB7 komputera oraz na dzielnik dekadowy. Po kolejnych podziałach sygnał podawany jest odpowiednio fx/10 na PB6, fx/100 na PB5 i fx/1000 na PB4. Do budowy wzmacniacza użyto tranzystorów impulsowych typu BSX 94, przy czym tranzystor T3 dobrano tak, by jego wzmocnienie prądowe było większe od 150. W przypadku braku tranzystorów impulsowych można w ich miejsce zastosować dowolne tranzystory npn m.cz. (BC 107, BC 528...) lub w.cz. (BF 194, BF 214...) zmieniając w razie konieczności obwód drukowany (rys. 2). Jako diodę D1 można zastosować dowolną krzemową diodę impulsową. Kondensatory, oprócz elektrolitycznych, powinny być ceramiczne, a pojemność kondensatorów blokujących może być większa od przyjętej. Przy zastosowaniu potencjometrów montażowych

1/2 UCY 7400

UCY 7490 UCY 7490 UCY 7490

BSX 94

widok od strony nóżek

widok od strony elementów

góra

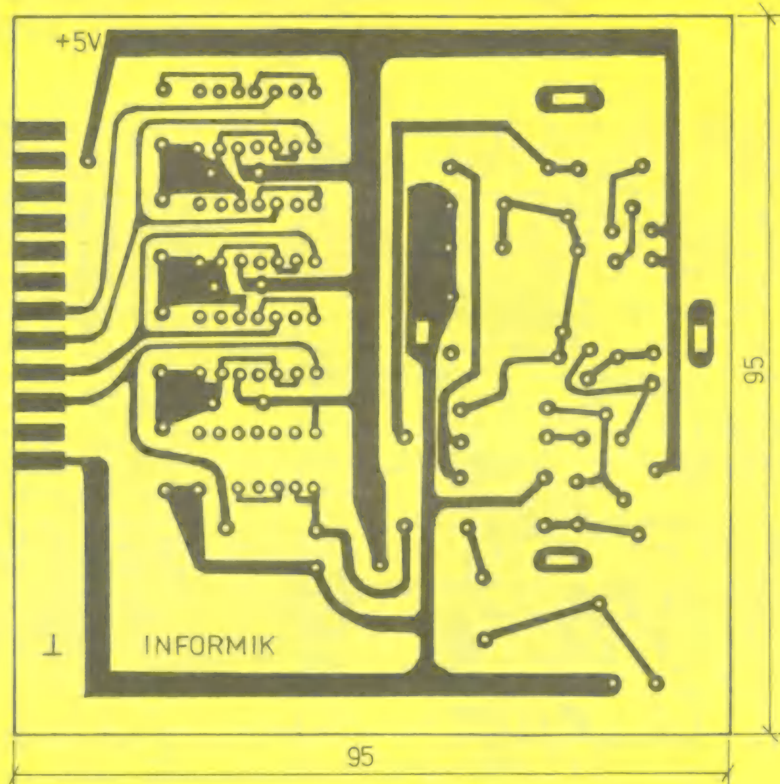
USER PORT

dół

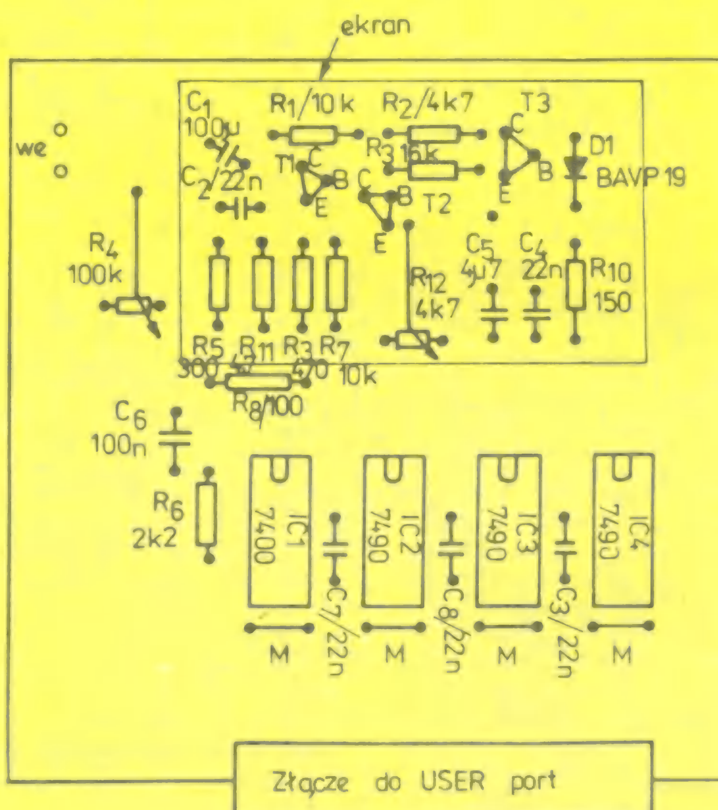
widok USER portu od strony tyłu komputera



Rys. 1. Schemat ideowy przystawki do pomiaru częstotliwości sygnału za pomocą komputera C-64



Rys. 2. Obwód drukowany przystawki



Rys. 3. Rozmieszczenie elementów na obwodzie drukowanym

```
10 rem czestosciomierz
20 poke53281,8:print chr$(58)
30 print chr$(147),"prosze czekac"
40 if peek(49152)<>169 or peek(49262)<>96 then go to 300
41 y=0:for x=0 to 4:y=y+peek(49152+x):next:if y=534 then go to 50
42 print"uwaga na port B komputera !!! ":poke49152,96:end
50 print chr$(147),chr$(14),chr$(159),chr$(18),
51 poke 781,6:poke782,12:poke783,0:sys65520:rem instrukcja plot
52 print"   Pomiar    "
60 rem automatyczny zakres pomiarowy
70 zp=4
80 poke 254,2^zp :rem dwa do potegi zp
90 poke251,0:poke252,0
100 sys 49152
110 f=peek(251)+256*peek(252)
120 if f >= 10000 then 230
130 if f<   1000 then 250
140 rem opracowanie wyniku
150 f=f*10^(7-zp)
160 if f=0 then kl=0 : goto 180
170 kl=int(log(f)/log(10)/3)
180 f=f/10^(3*kl)
190 jm$(0)="  Hz":jm$(1)="  kHz":jm$(2)="  mHz"
200 poke 781,10:poke782,12:poke783,0:sys65520:rem instrukcja plot
205 rem w lini 210 w cudzyslowie siedem spacji i siedem razy cofanie kursora
210 print "              ";f,jm$(kl)
220 goto 40
230 if zp<=4 then 140
240 zp=zp-1:goto 80
250 if zp>=7 then 140
260 zp=zp+1:goto 80
300 rem program maszynowy i jego loader
310 restore
320 for j=0 to 110
330 w=0
340 read b$
350 h=asc(b$):if h>57 then h=h-7
360 l=asc(right$(b$,1)):ifl>57 then l=l-7
370 w=16*(h-48)+l-48
380 poke49152+j,w
390 next j
400 y=0:for x=0 to 110:y=y+peek(49152+x):next:if y=16 189 then go to 50
410 print"pomylka w programie maszynowym":poke49152,96:end
500 data a9,00,8d,03,dd,ad,0f,dd,29,7f
510 data 8d,0f,dd,ad,0e,dd,09,80,8d,0e
520 data dd,ad,01,dd,25,fe,85,fd,a0,20
530 data a2,ff,ad,01,dd,25,fe,c5,fd,d0
540 data 08,ca,d0,f4,88,d0,ef,60,ea,78
550 data 20,60,c0,a9,00,aa,a8,8d,09,dd
560 data 8d,08,dd,ad,08,dd,c9,01,d0,f9
570 data 20,60,c0,e8,d0,01,c8,a9,00,cd
580 data 09,dd,b0,f2,cd,08,dd,b0,ed,86
590 data fb,84,fc,58,60,ea,ad,01,dd,25
600 data fe,d0,f9,ad,01,dd,25,fe,f0,f9,60

ready.
```

Program częstościomierza

OPIS PROGRAMU MASZYNOWEGO

```
., C000 A9 00    LDA #$00  -
., C002 8D 03 DD STA $DD03 - port B jako odbiornik
., C005 AD 0F DD LDA $DD0F -
., C008 29 7F    AND #$7F  -
., C00A 8D 0F DD STA $DD0F - bez alarmu
., C00D AD 0E DD LDA $DD0E -
., C010 09 80    ORA #$80  -
., C012 8D 0E DD STA $DD0E - czestotliwość sieci 50 Hz
., C015 AD 01 DD LDA $DD01 - pobranie portu B
., C018 25 FE    AND $FE   - stan wejścia aktualnie czynnego, maska z k. 254
., C01A 85 FD    STA $FD   - zapamietanie tego stanu w k. 253
., C01C A0 20    LDY #$20  - starszy bajt petli czasowej
., C01E A2 FF    LDX #$FF  - młodszy bajt petli czasowej
., C020 AD 01 DD LDA $DD01 - pobranie portu B
., C023 25 FE    AND $FE   -
., C025 C5 FD    CMP $FD   - czy sygnał na wybranym wejściu zmienił stan
., C027 D0 08    BNE $C031 - jest przebieg zmienny to możemy mierzyć
., C029 CA       DEX       -
., C02A D0 F4    BNE $C020 - czy mamy jeszcze czekać
., C02C 88       DEY       -
., C02D D0 EF    BNE $C01E - czy mamy jeszcze czekać
., C02F 60       RTS       - nie to powrót do BASIC-a
., C030 EA       NOP       -
., C031 78       SEI       - nie przeszkadzać-blokujemy przerwania
., C032 20 60 C0 JSR $C060 - zsynchronizowanie wykonywanego programu z wejściem
., C035 A9 00    LDA #$00  -
., C037 AA       TAX       -
., C038 A8       TAY       - szybko zerujemy dwubajtowy licznik impulsów -X,Y
., C039 8D 09 DD STA $DD09 - zerujemy licznik sekund
., C03C 8D 08 DD STA $DD08 - zerujemy licznik dziesiatych cześci sekundy
., C03F AD 08 DD LDA $DD08 -
., C042 C9 01    CMP #$01  -
., C044 D0 F9    BNE $C03F - zliczanie zaczniemy po pierwszej 0.1 sekundy
., C046 20 60 C0 JSR $C060 - rozpoznanie kolejnego impulsu na wejściu
., C049 E8       INX       -
., C04A D0 01    BNE $C04D -
., C04C C8       INY       - zliczenie go w dwubajtowym liczniku
., C04D A9 00    LDA #$00  -
., C04F CD 09 DD CMP $DD09 -
., C052 B0 F2    BCS $C046 -
., C054 CD 08 DD CMP $DD08 - czy upłyneła dokładnie jedna sekunda
., C057 B0 ED    BCS $C046 - nie to liczymy dalej
., C059 86 FB    STX $FB   - do k. 251 młodszy bajt zliczonych impulsów
., C05B 84 FC    STY $FC   - do k. 252 starszy bajt zliczonych impulsów
., C05D 58       CLI       - odblokowywujemy przerwania
., C05E 60       RTS       - powrót do BASIC-a
., C05F EA       NOP       -
., C060 AD 01 DD LDA $DD01 -
., C063 25 FE    AND $FE   -
., C065 D0 F9    BNE $C060 - czekamy na zero wczynnym kanale
., C067 AD 01 DD LDA $DD01 - jest
., C06A 25 FE    AND $FE   -
., C06C F0 F9    BEQ $C067 - było zero to czekamy na jedynke
., C06E 60       RTS       - było zero a po nim jedynka wiec powrót
```

Program maszynowy licznika impulsów

stojących lub o innych wymiarach trzeba poprawić obwód drukowany. Wzmacniacz wejściowy powinien być osłonięty ekranem (rys. 3). Najwięcej kłopotów sprawia złącze podłączeniowe do USER portu. Istnieją dwa sposoby rozwiązania tego problemu. Jeden to zakup odpowiedniego złącza na giełdzie (niestety cena jest wysoka), drugi to wykonanie złącza we własnym zakresie. Najlepiej w tym przypadku skleić pocięte na trzysegmentowe kawałki złącze szufladowe Eltry typu 802 064 o rastrze 3,75 mm tak, by co czwarty styk zachowywał raster 4 mm.

Uruchomienie przystawki

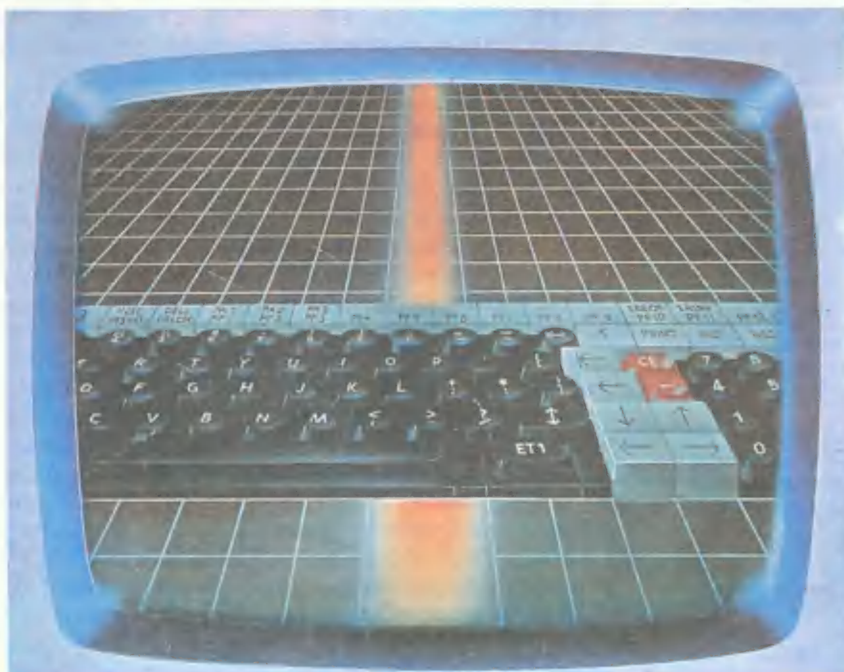
Po zmontowaniu pierwsze uruchomienie należy koniecznie wykonać bez podłączenia do komputera. Zapobiegnie to kosztownym skutkom naszych pomyłek. Gdy nie dysponujemy zasilaczem 5 V to układ możemy zasilać z baterii płaskiej 4,5 V. Pobór prądu z zasilacza musi być mniejszy od 120 mA. Wyjście komputera, z którego zasilany będzie układ ma dopuszczalne obciążenie 100 mA, obciążenie go prądem do 120 mA nie powoduje jednak żadnych problemów. Aby zmieścić się w 100 mA musielibyśmy zrezygnować z wzmacniacza wejściowego (20 mA) lub zastosować trudno dostępne elementy TTL o zmniejszonym poborze mocy serii 74LS... Gdy pobór prądu jest prawidłowy podajemy na wejście sygnał z generatora (można zastosować transformator sieciowy i dzielnik oporowy tak, by napięcie mniejsze było od 3,5 V). Kolejno na wyjściach wzmacniacza, przerzutnika, pierwszej, drugiej i trzeciej dekady kontrolujemy sygnał za pomocą oscyloskopu. W trakcie kontrolowania sygnału na wyjściu przerzutnika za pomocą potencjometru R12 ustawiamy przebieg o współczynniku wypełnienia równym 1/2. Gdy nie dysponujemy oscyloskopem sprawdzamy układ statycznie. Na katodzie diody D1 za pomocą potencjometru R12 ustalamy napięcie 1 V. Po czym na bazę tranzystora T1 przez opornik około 10 k podajemy na przemian sygnał o napięciu 0 V i napięciu 5 V względem masy układu kontrolując woltomierzem napięcie w odpowiednich punktach układu. Przy podaniu napięcia +5 V na katodzie diody powinno wystąpić napięcie większe od 2 V, a przy podaniu napięcia 0 V — mniejsze od 0,4 V. Przy takiej zmianie napięć przerzutnik powinien zmieniać stan wyjściowy z jedynki logicznej na zero. Gdy tak jest to kontrolujemy czy wyjścia (Qa, Qb, Qc, Qd) poszczególnych dekad zmieniają stan odpowiednio do podanej ilości impulsów. Jeżeli układ działa prawidłowo **sprawdzamy dokładnie czy odpowiednie sygnały trafiają na odpowiednie nóżki złącza** (opis wyprowadzeń USER portu na rys. 1) pamiętając o tym, że napięcie zasilania +5 V brane jest z drugiej strony łączówki (przy ewentualnych pomyłkach możemy uszkodzić komputer!). Jeżeli dysponujemy sprawdzonym i uruchomionym programem to przy wyłączonym komputerze dołączamy przystawkę do komputera, a następnie po jego załączeniu wczytujemy i uruchamiamy program.

Opis programu

Program napisany jest w języku BASIC. Po uruchomieniu ustalana jest barwa tła oraz znaków graficznych i wyświetlany napis: Proszę czekać. Następnie kontrolowana jest obecność właściwego programu maszynowego w obszarze od $C000 do $C06E, a w przypadku jego braku wpisywany jest tam za pomocą programu ładującego (linie 300—600). Po przepisaniu sprawdzana jest poprawność programu. W wypadku nieprawidłowych danych instrukcji data wypisywany jest komunikat i do komórki 49152 ($C000) wpisywany rozkaz powrotu RTS (kod $60). Gdy program maszynowy znajduje się we właściwym obszarze sprawdzamy czy na pewno port B, na który podajemy sygnały przełączany jest na odbiór (linia 41-42). W przeciwnym wypadku może bowiem dojść do uszkodzenia układu CIA obsługującego to wejście. Gdy wszystko jest w porządku wykonywana jest zasadnicza część programu. W komórce 254 ($FE) zapisujemy informację o tym, z którego wejścia portu B mamy zliczać impulsy, zerujemy dwubajtowy licznik impulsów zorganizowany w komórkach 251 i 252 ($FB i $FC) po czym zliczamy ilość impulsów, jaka z tego wejścia podana jest na licznik w ciągu jednej sekundy (linia 100). Z kolei sprawdzamy czy mieści się ona w przyjętym zakresie. Gdy nie, to o ile jest to możliwe zmieniamy zakres pomiarowy, a zatem i wejście (linie 120-130 i 230-260). Dysponując wynikiem pomiaru, normujemy jego postać (linia 140--180), przygotowujemy jednostki miary (linia 190), i w odpowiedniej formie wyświetlamy (linie 200-210). Po czym przystępujemy do ponownego pomiaru (linia 220).

Dużo trudności przy uruchamianiu programów publikowanych w czasopismach sprawia niewyraźny druk, dlatego w tym programie znaki edytora ekranowego zapisane są za pomocą instrukcji PRINT CHR$ () lub opisane słownie. W razie kłopotów z parametrami instrukcji DATA należy porównać je z kodem maszynowym. W pierwszej części programu licznika ($C000—$C02E) ustalamy odpowiednie warunki pracy układu CIA obsługującego port B, oraz sprawdzamy czy na ustalonym wejściu istnieje przebieg zmienny. Gdy nie stwierdzimy zmiany wartości wybranego bitu (maska z komórki $FE) oraz upłynie czas oczekiwania następuje powrót do BASIC-a. Układ CIA posiada szesnaście rejestrów. Dla nas istotne są: ośmiobitowy port B ($DD01), którego bity w zależności od stanu rejestru sterującego ($DD03) przyjmują lub wysyłają informację (jedynka na odpowiednim bicie rejestru sterującego oznacza, że ten bit w porcie B przygotowany jest do nadawania, zero — do odbioru), licznik dziesiętnych części sekundy ($DD08), licznik sekund ($DD09), oraz rejestry sterujące pracą zegara ($DD0E i $DD0F).

Gdy na wybranym wejściu jest przebieg mierzymy jego częstotliwość ($C031—$C06E). Synchronizujemy wykonywanie programu z przebiegiem wejściowym tak, by nie przegapić jakiegoś impulsu, zerujemy lokalny licznik impulsów (zorganizowany w rejestrach X, Y mikroprocesora ze względu na szybkość działania), zerujemy zegar i czekamy aż przebieg zmieni wartość z zera na jedynkę ($C060—$C06E). Tak długo pozostajemy w tym podprogramie jak długo nie wystąpi dodatnie zbocze przebiegu. Jeżeli w tym czasie zaniknie sygnał wejściowy to jedynym sposobem wyskoczenia z pętli jest przerwanie NMI (jednoczesne naciśnięcie przycisków RUN/STOP i RESTORE) lub wyzerowanie komputera RESET. Po dodatnim zboczu przebiegu zwiększamy licznik impulsów, kontrolujemy zegar (czas bramkowania) i w zależności od niego czekamy na kolejne dodatnie zbocze przebiegu lub kończymy zliczanie. Przed powrotem do BASIC-a zapisujemy w komórce $FB młodszy, a w komórce $FC starszy bajt ilości zliczonych impulsów w ciągu sekundy.



JANUSZ WRZEŚNIAK

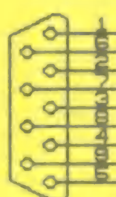
TRANSMITujmy!

Tekst ten nie jest bynajmniej próbą namawiania do powrotu do starego, poczciwego ZX Spectrum — urządzenie to, niechętnie nazywane przez profesjonalistów komputerem, zniknęło już niemal całkowicie z obszarów poważnych zastosowań. Postępująca żywiołowo w ubiegłych latach komputeryzacja polskich przedsiębiorstw pozostawiła w nich jednak sporo komputerków Sinclaira, a popularność ZX Spectrum w domach prywatnych wciąż opiera się konkurencji lepszych maszyn. Mimo niewątpliwie ograniczonych możliwości Spectrum warto pokusić się o lepsze wykorzystanie tego — posiadanego już — sprzętu, a zwłaszcza o zapewnienie przejścia między nim a aktualnie używanymi komputerami.

Z myślą o wszystkich użytkownikach ZX Spectrum, którzy w pracy przesiedli się już na komputery zgodne ze standardem IBM-PC opracowany został program TRANSMIT, służący do przesyłania plików tekstowych napisanych na ZX Spectrum w standardzie edytorów Poltasword i Tekst/Ed do plików IBM-PC w formacie popularnego Chi-Writera. Program ten może posłużyć do przeniesienia — bez żmudnego przepisywania — wszelkich opracowanych na ZX Spectrum dokumentacji, których wciąż sporo znajduje się w przedsiębiorstwach. Można też dzięki niemu przedłużyć aktywny żywot komputerków Sinclaira, stosując je jako stacje przygotowywania danych dla IBM-a (pliki tekstowe mogą zawierać także np. teksty programów lub dane dla arkuszy obliczeniowych i baz danych...). Program TRANSMIT

Wydruk 1: Program transmisyjny (część w BASIC-u)

```
10 RANDOMIZE USR 40003
20 CLS : PRINT #0;" TEKST Z DYSKU CZY Z TAŚMY ?"
30 PAUSE 0: LET A$=INKEY$
40 IF A$<>"D" AND A$<>"d" THEN  GO TO 120
50 CLS : PRINT #0;" Włóż dyskietkę z tekstem ''
   " i naciśnij dowolny klawisz ": PAUSE 0
60 CLS : CAT *
70 INPUT "KTÓRY KATALOG ? "' LINE A$
80 IF A$<>"" THEN  GO TO *A$: CAT *
90 INPUT " KTÓRY PLIK ? "',LINE A$
100 IF A$="" THEN  BEEP 0.5,5: GO TO 100
110 LOAD *A$CODE 45000: GO TO 150
120 IF A$<>"T" AND A$<>"t" THEN  GO TO 10
130 INPUT " KTÓRY PLIK ? "' LINE A$
140 LOAD A$CODE 45000
150 CLS : INPUT " POD JAKĄ NAZWĄ ZAPISAĆ ? "' LINE X$
160 RANDOMIZE USR 40000
170  FOR  I=1  TO  5:  BEEP  0.05,40:  BEEP  0.1,60:  BEEP  0.1,50:
    NEXT I
180 CLS : PRINT #0;" PRZESYŁAMY DALEJ (T/N) ? "
190 PAUSE 0: LET A$=INKEY$
200 IF A$="T" OR A$="t" THEN  GO TO 10
210 RANDOMIZE USR 40006: CLOSE #*: STOP
220 BORDER 0: PAPER 0: INK 7: BRIGHT 1: CLEAR 39999
230 LOAD *"TRANSMIT.EXE"CODE 40000
240 PRINT " SFORMATUJ KANAŁ SZEREGOWY "'
    " WEDŁUG WZORU: "''
250 PRINT "Text / Bytes : Bytes"
260 PRINT "XON/XOFF: No"
270 PRINT "Input with wait: No"
280 PRINT "Baud rate: 0"
290 PRINT "Parity: None"
300 PRINT "Stop bits: A"
310 PRINT "Bits/char: D"
320 FORMAT *":ch_a.scp"
330 OPEN #*3;":ch_a.scp";o
340 GO TO 10
350 SAVE *"TRANSMIT" LINE 220
```



Wydruk 2: Loader kodu maszynowego

```
  10 CLEAR 39999
  20 RESTORE 110
  30 LET s=0: LET adr=40000
  40 FOR i=1 TO 974
  50 READ x: POKE adr,x
  60 LET s=s+x: LET adr=adr+1
  70 NEXT i
  80 IF s<>85772 THEN  PRINT  FLASH 1;"BŁĄD W DANYCH !!":
     STOP
  90 CLS : INPUT "WŁÓZ DYSKIETKĘ I NACIŚNIJ ENTER"; LINE A$
 100 SAVE *"TRANSMIT.EXE"CODE 40000,974
 110 DATA 195,170,156,195,73,156,195,87,156,33
 120 DATA 200,175,17,201,175,1,255,79,54,32
 130 DATA 237,176,201,205,120,159,13,10,69,67
 140 DATA 72,79,32,79,70,70,13,10,67,76
 150 DATA 83,13,10,69,67,72,79,32,80,76
 160 DATA 73,75,73,32,80,82,90,69,83,76
 170 DATA 65,78,69,32,90,69,32,83,80,69
 180 DATA 67,84,82,85,77,58,13,10,68,73
 190 DATA 82,32,42,46,67,72,73,13,10,71
 200 DATA 79,84,79,32,69,88,73,84,13,10
 210 DATA 26,0,175,211,254,201,33,199,255,1
 220 DATA 0,80,126,254,32,32,7,43,11,120
 230 DATA 177,200,24,244,197,6,8,33,11,157
 240 DATA 54,32,35,16,251,205,239,159,35,126
 250 DATA 35,35,17,11,157,1,8,0,254,8
 260 DATA 48,7,19,11,60,254,8,32,249,237
 270 DATA 176,205,120,159,13,10,77,79,68,69
 280 DATA 32,67,79,77,49,58,57,54,48,48
 290 DATA 44,78,44,56,44,49,13,10,67,76
 300 DATA 83,13,10,67,79,80,89,32,67,79
 310 DATA 77,49,32,32,32,32,32,32,32,32
 320 DATA 32,46,67,72,73,13,10,84,82,65
 330 DATA 78,83,13,10,26,0,33,11,157,17
 340 DATA 178,158,1,8,0,237,176,205,107,13
 350 DATA 62,2,205,1,22,205,226,159,22,10
 360 DATA 1,32,32,32,32,32,32,32,32,78
 370 DATA 97,99,105,115,110,105,106,32,69,78
 380 DATA 84,69,82,13,13,103,100,121,32,73
 390 DATA 66,77,32,115,107,111,110,99,122,121
 400 DATA 32,111,112,101,114,97,99,106,101,32
 410 DATA 100,121,115,107,111,119,101,22,14,4
 420 DATA 105,32,110,97,112,105,115,122,101,58
 430 DATA 32,67,79,80,89,32,67,79,77,49
 440 DATA 32,46,46,46,0,175,50,8,92,62
 450 DATA 5,211,254,118,58,8,92,183,40,249
 460 DATA 6,20,118,16,253,205,107,13,62,2
 470 DATA 205,1,22,205,226,159,22,3,1,139
 480 DATA 131,131,131,131,131,131,131,131,131,131
 490 DATA 131,131,131,131,131,131,131,131,131,131
 500 DATA 131,131,131,131,131,131,131,131,135,22
 510 DATA 4,1,138,32,32,32,32,80,114,111
 520 DATA 103,114,97,109,32,100,111,32,116,114
 530 DATA 97,110,115,109,105,115,106,105,32,32
 540 DATA 32,133,22,5,1,138,32,32,32,32
 550 DATA 32,32,112,108,105,107,111,119,32,32
 560 DATA 116,101,107,115,116,111,119,121,99,104
 570 DATA 32,32,32,32,133,22,6,1,138,32
 580 DATA 32,122,32,84,65,83,87,79,82,68
 590 DATA 65,32,100,111,32,67,72,73,45,87
 600 DATA 82,73,84,69,82,65,32,133,22,7
 610 DATA 1,138,32,32,32,32,32,32,32,32
 620 DATA 32,32,32,32,32,32,32,32,32,32
 630 DATA 32,32,32,32,32,32,32,32,32,32
 640 DATA 133,22,8,1,138,32,32,32,32,32
 650 DATA 32,32,32,32,74,97,87,32,127,32
 660 DATA 49,57,56,56,32,32,32,32,32,32
 670 DATA 32,32,32,133,22,9,1,142,140,140
 680 DATA 140,140,140,140,140,140,140,140,140,140
 690 DATA 140,140,140,140,140,140,140,140,140,140
 700 DATA 140,140,140,140,140,140,141,22,14,4
 710 DATA 87,89,83,89,76,65,77,32,84,69
 720 DATA 75,83,84,32,68,79,32,80,76,73
 730 DATA 75,85,32,22,16,10,32,32,32,32
 740 DATA 32,32,32,32,46,67,72,73,0,62
 750 DATA 3,211,254,193,33,63,0,9,6,6
 760 DATA 203,28,203,29,16,250,62,1,164,103
 770 DATA 229,205,120,159,13,10,0,209,33,200
 780 DATA 175,6,64,175,50,133,159,213,126,254
 790 DATA 128,56,2,62,32,254,32,56,5,205
 800 DATA 134,159,24,15,205,59,159,50,1,159
 810 DATA 205,120,159,92,50,0,92,49,0,35
 820 DATA 16,222,58,133,159,230,128,40,5,62
 830 DATA 32,205,174,159,205,120,159,13,10,92
 840 DATA 43,13,10,0,209,27,122,179,40,2
 850 DATA 24,185,205,120,159,92,44,13,10,92
 860 DATA 45,13,10,92,61,13,10,26,0,175
 870 DATA 211,254,201,229,197,71,33,82,159,35
 880 DATA 126,35,184,40,9,254,255,32,246,62
 890 DATA 32,193,225,201,126,24,250,15,97,24
 900 DATA 99,25,101,26,108,27,110,28,111,29
 910 DATA 115,30,120,31,122,0,65,1,67,2
 920 DATA 69,3,76,4,78,5,79,7,83,10
 930 DATA 88,11,90,255,227,126,35,183,40,5
 940 DATA 205,174,159,24,246,227,201,0,229,33
 950 DATA 133,159,203,126,40,11,254,32,32,2
 960 DATA 203,190,205,174,159,225,201,254,32,32
 970 DATA 12,62,92,205,174,159,62,32,205,174
 980 DATA 159,225,201,203,254,24,247,0,245,58
 990 DATA 173,159,47,230,7,50,173,159,211,254
1000 DATA 241,253,229,243,253,33,0,0,205,8
1010 DATA 0,253,225,229,33,0,32,34,71,33
1020 DATA 119,33,1,0,34,69,33,62,3,50
1030 DATA 46,33,205,26,6,225,205,3,6,201
1040 DATA 227,126,183,40,6,229,215,225,35,24
1050 DATA 246,227,201,62,88,230,31,79,203,241
1060 DATA 42,75,92,126,230,127,40,14,185,32
1070 DATA 3,23,135,240,197,205,184,25,235,193
1080 DATA 24,237,207,1
```

Wydruk 3: Procedury wsadowe IBM'a

● *procedura TRANS.BAT*

```
cls
echo off
erase trans*.bat
mode   com1:9600,n,8,1
copy com1 trans1.bat
copy trans1.bat+trans2.bat transmit.bat
transmit
```

● *procedura TRANS2.BAT*

```
trans
exit
```

przyda się też wszystkim posiadaczom ZX Spectrum, którzy pracują na nim w domu, a z rezultatów chcą korzystać w pracy.

Program TRANSMIT wymaga ZX Spectrum 48K (lub Plus), wyposażonego w popularny interfejs dyskowy firmy Timex (oferowany w wersji oryginalnej) przez Centralną Składnicę Harcerską i Baltonę, a jako kopie — przez firmę Polbrit). Interfejs ten oprócz stacji 3-calowych dysków posiada także dwa łącza szeregowe standardu RS-232. Jedno z tych łączy zostało wykorzystane do przesyłania danych do IBM-a.

Program TRANSMIT składa się z dwóch fragmentów — w języku BASIC (wydruk 1) i kodzie maszynowym. Przedstawiony na wydruku 2 program ładujący tworzy blok kodu, który należy zapisać na tej samej dyskietce, co program w BASIC-u.

Na komputerze docelowym (IBM-PC/XT/AT lub zgodnym) należy utworzyć osobny podkatalog — np. pod nazwą TRANS. W katalogu tym należy umieścić dwie procedury wsadowe: TRANS.BAT i TRANS2.BAT. Procedurom tym należy następnie nadać atrybut „tylko do odczytu" — przy pomocy polecenia systemowego.

attrib +r *.bat

Komputer IBM łączymy ze interfejsem Timex za pomocą odpowiedniego przewodu wielożyłowego dołączonego do kanału A stacji Timex i portu COM1 komputera IBM (zwykle port ten znajduje się na karcie Multi I/0). Transmisję rozpoczynamy wczytując do ZX Spectrum program TRANSMIT, a na IBM-ie — wydając (w podkatalogu TRANS) polecenie TRANS.

Program TRANSMIT rozpoczyna działanie od ustalenia parametrów transmisji szeregowej (tzw. formatowania kanału). W odpowiedzi na zadawane przez komputer pytania należy naciskać klawisze zgodnie z wypisaną na ekranie „ściągawką". Po zakończeniu formatowania program wchodzi w główną pętlę, w której odpowiadamy na pytania o źródło (dysk czy kaseta) pliktu tekstowego, nazwę podkatalogu w systemie TOS i nazwę pliku tekstowego do wysłania. Podajemy też nazwę, pod jaką plik zostanie zapisany na dysku IBM-a.

UWAGA! Z podanej nazwy program uwzględnia tylko pierwsze 8 znaków i automatycznie dodaje rozszerzenie. .CHI. Plik jest zapisywany zawsze w bieżącym katalogu — tj. TRANS. Podanie nazwy pliku istniejącego już w tym katalogu spowoduje skasowanie dotychczasowej zawartości pliku. Pożądane jest zatem, by przed każdą sesją transmisyjną kasować wszystkie pliki z katalogu TRANS (oczywiście poza TRANS.BAT i TRANS2.BAT, które są zabezpieczone przed skasowaniem).

Po podaniu nazwy pliku docelowego program wysyła do IBM-a komunikat o pliku docelowym. IBM przetwarza ten komunikat i przygotowuje się do przyjęcia właściwego pliku, co jest sygnalizowane wyczyszczeniem ekranu i napisem

⟨*prompt*⟩ COPY COM1 ⟨*nazwa_pliku*⟩ .CHI

Należy wówczas wcisnąć dowolny klawisz ZX Spectrum, co rozpocznie transmisję pliku tekstowego. Po jej zakończeniu IBM samoczynnie rozpocznie oczekiwanie na kolejną transmisję, a ZX Spectrum zapyta, czy przesyłać dalej. Odpowiedź negatywna spowoduje wysłanie do IBM-a komunikatu kończącego sesję i zakończenie pracy programu. IBM wypisze wówczas spis plików odebranych ze ZX Spectrum (pliki z rozszerzeniem .CHI) i zakończy wykonywanie procedur wsadowych.

Przesyłany tekst jest przekodowywany do postaci akceptowanej przez Chi-Writera, z zachowywaniem formatu tekstu i polskich znaków. Znaki sterujące zawarte w tekście (np. włączanie i wyłączanie druku o podwójnej wysokości/szerokości itp.) są ignorowane, tj. zamieniane na spacje. Wysyłane wiersze mają tzw. miękkie końce, co z jednej strony pozwala łatwo przeformatować pod Chi-Writerem teksty pisane z dowolnie ustawionymi marginesami, ale zmusza zarazem do ręcznego wprowadzenia „twardych" końców wierszy po każdym akapicie — nieostrożna próba sformatowania świeżo przesłanego tekstu spowoduje sklejenie wszystkich akapitów, tytułów itd. w jeden ciągły tekst.

W przypadku wczytywania tekstów z kasety może się zdarzyć błąd odczytu i przerwanie pracy programu. Należy wówczas uruchamiać program instrukcją GOTO 10. Ponowne uruchomienie programu po omyłkowym zakończeniu pracy powinno się odbywać instrukcją GOTO 330, aby uniknąć ponownego formatowania kanału szeregowego.

Od redakcji:

Opisane rozwiązanie zostało sprawdzone w praktyce i od półtora roku wykazuje się swoimi zaletami. Opis powyższy wypada jednak uzupełnić pewnymi uwagami o charakterze eksploatacyjnym.

Stacja dysków 3" TIMEX wyposażona jest w bardzo złej jakości złącze szufladowe. W trzech eksploatowanych urządzeniach po okresie nieużywania występowały kłopoty z zapoczątkowaniem prawidłowej transmisji, które ustępowały po oczyszczeniu styków złącza szufladowego. Jest to niestety jedyna droga dla tych, którzy nie chcą rozbierać urządzenia w celu wymiany złącza na lepsze.

Ze względu na objętość nie został zamieszczony listing programu źródłowego; jeżeli jednak Czytelnicy uznają, że jego zamieszczenie jest wskazane (prosimy o listy!), obiecujemy uczynić to w jednym z najbliższych numerów.

CIEKAWE KSIĄŻKI

JĘZYKI MIKROKOMPUTERÓW (Przewodnik dla początkujących). Praca zbiorowa pod redakcją Mike'a Ducka. Wydawnictwa Naukowo-Techniczne 1988, wydanie I, nakład 40 000 egz.

W pierwszej chwili, gdy porównuje się szumny tytuł książki z jej skromną objętością, ogarnia nas zdumienie i niedowierzanie — czy w tak małej książeczce można było zawrzeć tak wiele rzetelnej informacji? Wszelako na naszym rynku książkowym brak jest takiej książki o charakterze porównawczym. Zajrzyjmy na drugą stronę okładki — czytamy tam:

„Autorzy tej popularnej książki (...) wprowadzają młodzież w tajniki programowania mikrokomputerów. Omawiają w przystępny sposób sześć najpopularniejszych języków programowania: BASIC, Pascal, Logo, Prolog, Comal i Forth, a także je porównują."

No cóż — określenie „najpopularniejszy" mało pasuje do egzotycznego u nas Comalu. Gdybyż zamiast języka Comal był opisany język C ... Zajrzyjmy do wstępu. Mike Duck pisze:

„... wszystkie języki były tematem wieczorowych kursów dla początkujących, które zorganizował Wydział Elektroniki i Łączności Politechniki Londyńskiej. (...) Ponieważ kursy cieszyły się ogromną popularnością, opis sześciu omawianych języków postanowiłem ze-

brać w jednej książce, przeznaczonej dla początkujących".

Dodajmy jeszcze, że oryginał książki ukazał się w Wielkiej Brytanii w 1984 roku.

Znając już genezę książki i wiedząc, do kogo jest adresowana, postanowiłem przeprowadzić pewien test. Zaprosiłem do współpracy syna jednego z moich przyjaciół, kilkunastolatka stawiającego pierwsze kroki w dziedzinie informatyki, dając mu do dyspozycji książkę, dwa komputery (ZX Spectrum i IBM-PC — bo z takimi miał już do czynienia), oprogramowanie i siebie w charakterze pomocy technicznej, a w zamian otrzymując prawo opisania efektów nauki w oparciu o tę pozycję literatury. Poniżej krótko opisuję przebieg doświadczenia.

1. BASIC. Okazało się, że mój młody przyjaciel, jako posiadacz komputera ośmiobitowego, zna BASIC (a dokładniej: jeden z jego dialektów) wręcz doskonale. Szybko przekartkował rozdział bezbłędnie wyłapując różnice między znanym mu dialektem, a wersją opisaną w książce. Zauważmy, że dla posiadacza takiego komputera ze względu na obecność interpretera tego języka w pamięci stałej (Czytelnik zechce mi darować pewne generalizowanie faktów) opanowanie go jest przykrą, acz niezbędną koniecznością. I tu da o sobie znać problem różnic poszczególnych dialektów. Na szczęście autor rozdziału przewidział to, zamieszczając na samym początku krótką notkę objaśniającą.
2. PASCAL. Jako narzędzie tekstu posłużyły: Turbo Pascal 4.0 i Hisoft Pascal 4TM16. Pomijając kłopoty z edytorem i wprowadzeniem tekstu znowu dały o sobie znać różnice w dialektach, na szczęście bardzo drobne. To, co sprawiło największy kłopot mojemu młodemu przyjacielowi, to problemy typów danych (opisanych rzeczywiście nad wyraz lakonicznie), i programowania strukturalnego. Wydaje się, że ten problem jest zbyt lekko potraktowany w tej publikacji, szczególnie, że poprzedni rozdział traktował o języku, który niewiele ma z programowaniem strukturalnym wspólnego.
3. LOGO. Okazało się, że Logo nie jest całkowicie obce mojemu współtestującemu. Przedmiotem testu było PTI LOGO (tłumacz uwzględnił w książeczce polską wesję tego języka — brawo!). Ogólne uwagi nasuwające się po zakończeniu tekstu:
   a) w rozdziale za dużo nacisku położono na grafikę, a za mało na obliczenia numeryczne, do czego Logo też może służyć;
   b) autor duży nacisk położył na problemy rekurencji w Logo, za co należy mu się uznanie;
   c) zamieszczone zostało zdjęcie żółwia-robota, obiektu rzadko u nas spotykanego.
4. COMAL. Testu nie przeprowadziliśmy ze względu na brak oprogramowania. Mojemu młodemu przyjacielowi Comal, będący czymś pośrednim pomiędzy BASIC-em a Pascalem, wydał się wydumany, a sensowność jego opisu — wątpliwa.
5. PROLOG, a dokładniej Micro Prolog. Testowaniu podlegała (zgodnie z zaleceniami autora rozdziału) wersja na ZX Spectrum. I tu reakcje mojego współpracownika okazały się na tyle charakterystyczne, że opiszę je dokładniej.

Wgłębiwszy się we wstęp przeczytał on mniej więcej pół rozdziału nie dotykając komputera. Następnie wrócił na sam początek i zaczął „wklepywać" przykładowe listy i klauzule. Mniej więcej przez pięć stron szło jak po maśle, ale nagle w pewnym momencie komputer zaczął reagować komunikatem błędu. Porównanie echa na ekranie z przykładami w książce wykazało, że przykłady wprowadzone były bezbłędnie. Gwałtowna próba znalezienia erraty do książki zakończyła się jedynie oderwaniem słabo przyklejonej kartonowej okładki. Głębsze przemyślenia naprowadziły go wreszcie na myśl, że kłopoty jego spowodowane były pewnymi niekonsekwencjami deklaracji klauzul.

I na tym, niestety, zakończyliśmy testy, bowiem przygoda z Prologiem na tyle zniechęciła mojego młodego przyjaciela, że wymówił się od testowania rozdziału poświęconego językowi Forth. Jestem więc zdany jedynie na siebie. Moja opinia o tym rozdziale jest pozytywna. Należy pamiętać, że Forth jest językiem, którego struktura bardzo znacznie różni się od innych. Opis struktury języka jest klarowny i rzeczowy. Jedynie ostatnie podrozdziały, traktujące o użyciu drukarki kreślącej stanowiącej wyposażenie mało u nas popularnego komputera SHARP MZ 700, są mało przydatne w naszych warunkach. Nieco deprymujący jest również fakt niepodania literatury o tym języku.

Reasumując: uważam, że książka może być pozycją wartościową dla młodego adepta wiedzy informatycznej, ale tylko przy spełnieniu następujących warunków:

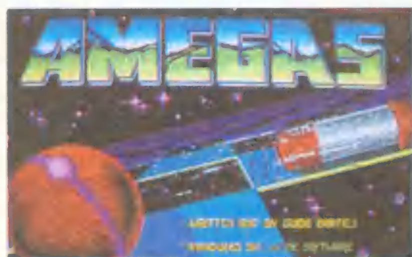
1. książka ta NIE BĘDZIE TRAKTOWANA JAKO PODRĘCZNIK, a jedynie jako przewodnik o charakterze informacyjno-porównawczym;
2. dobrze podczas czytania tej pozycji mieć możliwość zwrócenia się do fachowca (w celu rozproszenia wątpliwości), albo przynajmniej mieć w zasięgu ręki rzetelne podręczniki programowania w tych językach.

Bardzo cenne jest umieszczenie przez tłumacza spisu literatury polskiej (niestety — bez literatury dotyczącej języka Forth).

Na zakończenie kilka ogólników: książka wydana jest starannie, wydrukowana wyraźnie na dobrej klasy papierze. Pewne niezadowolenie może budzić bardzo niska jakość reprodukowanych zdjęć, a także mała trwałość połączenia okładki z resztą książki (patrz p. 5). Również cena (550 zł) jako cena książki popularnej, adresowanej do młodzieży wydaje się nieco za wysoka.

Dariusz Adam Przygoda

CO NOWEGO Z „AMIGĄ"



Obecność tego legendarnego już komputera na krajowym rynku użytkowników komputerów stała się faktem. Niestety nie miał on do tej pory żadnego związku z obecnością na łamach krajowych czasopism poświęconych informatyce. Jak sprawdziliśmy (o czym dalej), powstało zupełnie spontanicznie kilka klubów użytkowników Amigi liczących łącznie kilkaset osób. Przypuszczać należy, że liczba faktycznie znajdujących się na naszym rynku Amig jest znacznie większa. Specyficzną cechą polskiego rynku komputerowego jest chroniczny brak literatury i dokumentacji w chwili pojawienia się nowego (u nas) komputera. Oprogramowania, również w tym przypadku, jest pod dostatkiem. Chcąc wypełnić tę lukę, wszystkich użytkowników Amigi (zrzeszonych i nie ujawnionych) informujemy, że zamierzamy opublikować na łamach „InforMika" cykl artykułów o tematyce: „Prezentacja Amigi", „Co Amiga potrafi", „Grafika", „Muzyka".

Oczekujemy dalszych propozycji ze strony naszych przyszłych czytelników, a w szczególności klubów.

Nie chcąc narzucać jakichkolwiek form organizacyjnych ruchowi klubowemu, publikujemy jedynie adresy klubów dla wszystkich czytelników mających ochotę nawiązać bliższą współpracę.

Kraków:
Commodore Amiga Club
Piotr Podermański, 2 Pułku Lotn. 10/40
31-867 Kraków
Marek Chyla, Os. Kolorowe 9/16
31-939 Kraków
Klub wydaje własny biuletyn, zorganizował dwa ogólnopolskie zjazdy, liczy ok. 100 członków.

Opole:
Amiga Commodore Club
Przemysław Koziarski, ul Pasieczna 4a/10
45-087 Opole
Klub zorganizował dwa ogólnopolskie zjazdy, liczy ok. 30 członków.

Warszawa:
Klub użytkowników mikrokomputera Amiga „AMIGOS" ul. Potockich 111
Warszawa
Klub liczy ok. 40 członków.



KRZYSZTOF WIŚNIEWSKI

OPTYCZNE PAMIĘCI DYSKOWE

Od wielu lat w laboratoriach naukowych interesowano się problemem optycznej rejestracji i odczytu danych cyfrowych. Jak to zwykle bywa, zastosowania pojawiły się tam, gdzie można było spodziewać się maksymalnie szybkiego zwrotu nakładów finansowych. Pierwszymi zastosowaniami optycznego odczytu nie były pamięci masowe komputerów, ale domowe odtwarzacze sygnałów fonicznych i wizyjnych. Zatem nie wysoko wyspecjalizowane zastosowania profesjonalne, ale urządzenia typu „Compat Disc" i „Laser Vision" zdominowały pierwotne kierunki rozwoju zapisu optycznego.

Początkowo na przeszkodzie rozpowszechnieniu optycznych pamięci komputerowych stanęła możliwość zastosowania ich tylko jako pamięci typu ROM (*Read Only Memory*) — czyli tylko do odczytu. Przy pojemnościach setek megabajtów do kilku gigabajtów problemem nie jest produkcja płyty, ale jej zapełnienie informacją. W tak olbrzymiej przestrzeni adresowej można zapisać spory księgozbiór. Pamiętać należy, że zwykła strona maszynopisu pochłania od dwu do trzech kilobajtów pamięci. Jeżeli średnio książka z naszej bibliotek będzie mieć 500 stron, to na takiej płycie zmieści się od 500 do 1000 książek. Samo wprowadzenie takiej ilości danych do komputera podczas pierwotnego zapisu kosztuje bardzo drogo. Opłaca się to tylko dla wielkich serii wydawniczych takich, jak encyklopedie, mapy, leksykony.

Jednym z pierwszych zastosowań było wydanie Biblii w oryginale i kilkujęzycznym tłumaczeniu. Przy kilkuset milionach wierzących można liczyć na odpowiedni zbyt.

Następnym krokiem były płyty z jednorazowym zapisem danych. Funkcjonalnie odpowiadają one pamięciom stałym programowalnym typu PROM. Istota zapisu polega na odparowywaniu za pomocą lasera cienkiej warstwy metalu naniesionej na powierzchnię nośnika szklanego lub plastikowego. Po zapisaniu danych płyta pamięciowa wygląda podobnie do klasycznego dysku CD. Pojemność jest praktycznie tego samego rzędu co płyt stałych, programowanych przez producenta i jest tysiące razy większa od największych półprzewodnikowych pamięci PROM. Ciągle jednak poszukiwano sposobu swobodnego zapisu i odczytu pamięci optycznych. I tym razem niezastąpiona okazała się technika laserowa i wcześniej już opracowana metoda odczytu płyty CD. Otóż płyta optyczna wielokrotnego zapisu i odczytu pod względem funkcjonalnym odpowiada całkowicie zapisowi magnetycznemu na dyskach elastycznych lub twardych.

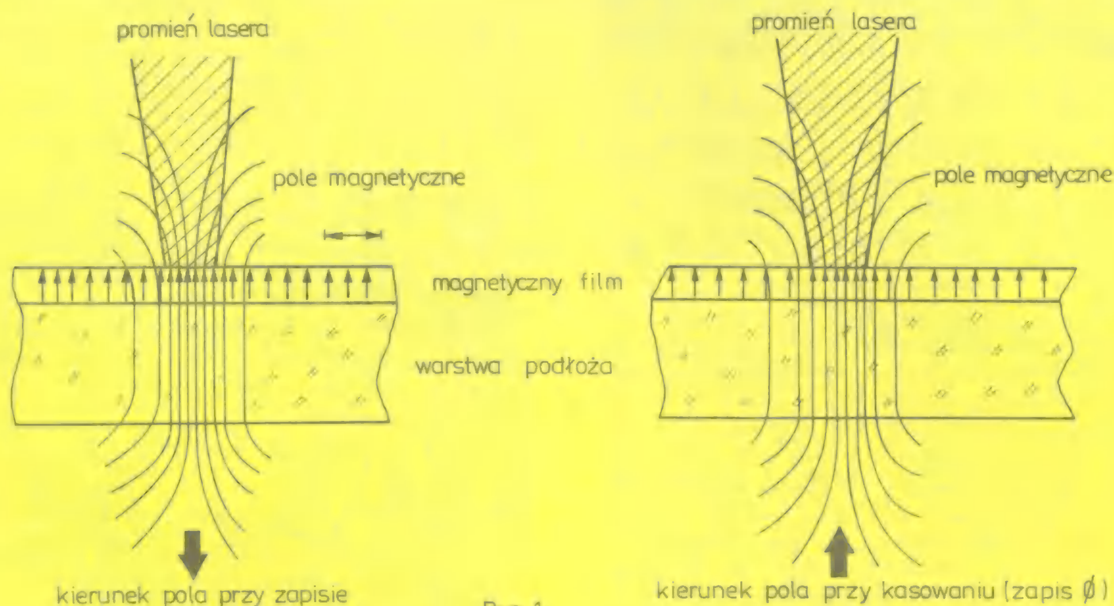
Całkowicie nowy jest jednak sposób zapisu i kasowania informacji. Zastosowano znane w fizyce zjawisko magnetooptyczne. Podobnie jak w poprzednio opisanych dyskach CD na krążek szklany lub plastikowy nanosi się bardzo cienką warstwę metalu. Jej grubość wynosi od 0,01 do 0,1 μm, co pozwala zminimalizować energię potrzebną do zapisu i odczytu danych. Metal też nie jest dowolny, ale o najlepszych właściwościach magnetooptycznych. Poszukiwania takiego nośnika uwieńczone zostały sukcesem w laboratoriach firmy IBM. Okazało się, że stop gadolin-kobalt jest najbardziej odpowiednim materiałem wykazującym właściwości magnetooptyczne.

Materiał ten zawiera w sobie mikroskopijne obszary magnetyczne zorientowane, wykazujące na dodatek stałą w czasie orientację magnetyczną. Obraz mikroskopowy takiej warstwy przypomina rozrzucone chaotycznie i zastygnięte w bryle plastiku miniaturowe magnesy. W tym stanie nośnik nie przedstawia dla nas ciekawych właściwości fizycznych. Interesujące rzeczy zaczynają się dziać dopiero po umieszczeniu materiału w polu magnetycznym i podgrzaniu do temperatury około 200°C. W tej temperaturze, nazywanej punktem Curie, następuje radykalna zmiana właściwości magnetycznych materiału, jednak bez zmiany stanu skupienia. Zjawisko to nazywane jest przejściem fazowym i odnosi się do radykalnych zmian cech fizycznych materiału w obrębie jednego stanu skupienia.

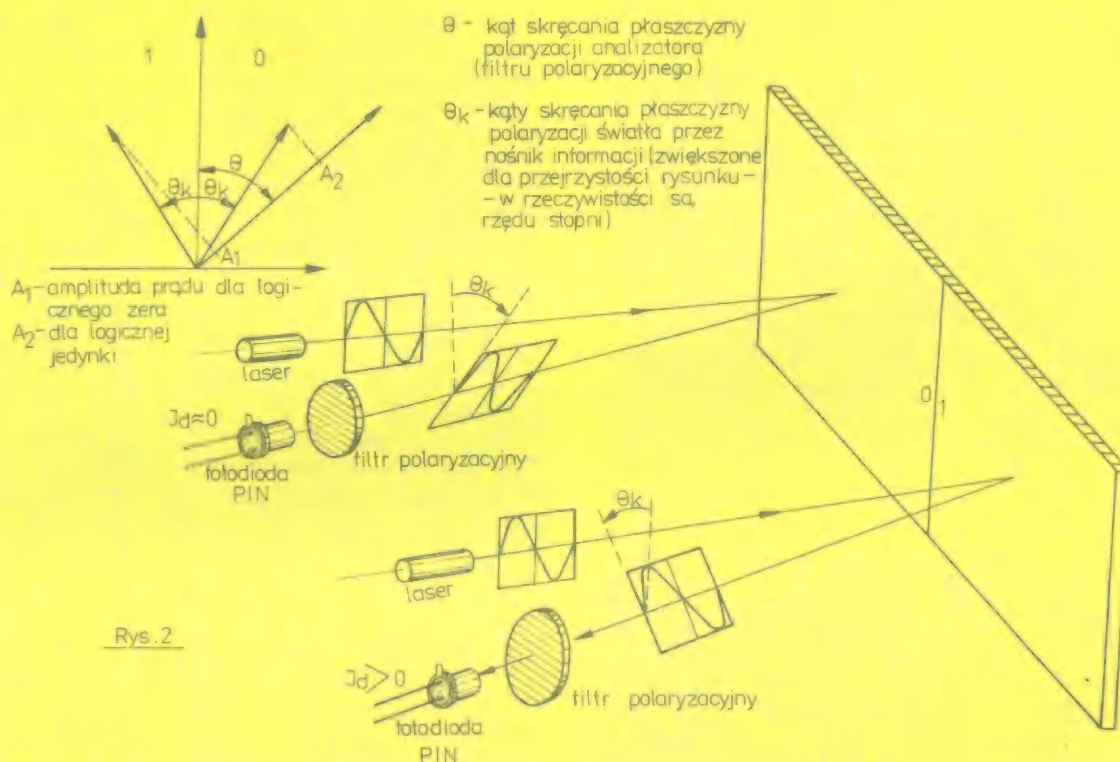
Okazało się, że przy zastosowaniu stopu gadolin-kobalt o wyżej wspomnianej grubości całkowicie wystarcza do zapisu i kasowania energia około 0,3 do 1 nJ. Pozwala to na dużą szybkość zapisu wynoszącą od 25 do 100 nr/bit., oraz szybkość transmisji danych wynoszącą 5 Mbaud. Proces zapisu przedstawiony jest na rys. 1. Najpierw do obszaru dużo większego niż zajmuje pojedynczy bit informacji, zostaje przyłożone pole magnetyczne, a następnie jednym błyskiem laserowym podgrzewamy obszar o średnicy 2 μm (odpowiada to jednemu bitowi), w którym następuje polaryzacja magnetyczna nośnika. Kierunek tej polaryzacji określony jest przez kierunek przyłożonego pola magnetycznego.

Tak zapisaną płytę można by odczytać za pomocą zwykłej głowicy magnetycznej, gdyby nie mikroskopijne wymiary pojedynczego bitu (tylko 2 μm). I znowu ratuje nas z opresji technika laserowa. Okazuje się, że obszary o kierunku polaryzacji północnym lub południowym powodują skręcają płaszczyznę polaryzacji odbitego od nich światła w przeciwne

Dokończenie na IV str. okł.



Rys.1



Rys.2

UWAGA CZYTELNICY!

W artykule Rolanda Wacławka „BUDZIK BEZ TRYBIKÓW I SPRĘŻYNEK" w „InforMiku" I/89 omyłkowo zamieszczono dwa razy ten sam listing w różnej skali. Jakkolwiek nie obniża to w żaden sposób wartości merytorycznej artykułu, za redakcyjne niedopatrzenie Autora i naszych Czytelników serdecznie przepraszamy!

KOMPUTER W SZKOLE

ALGORYTM WYZNACZANIA WSPÓŁRZĘDNYCH SŁOŃCA, KSIĘŻYCA I PLANET

CZĘŚĆ I

Umiejętność określania pozycji ciał niebieskich na sferze odgrywa istotną rolę w astronomii obserwacyjnej. Wprawdzie dokładne położenie Słońca, Księżyca i planet można odszukać w kalendarzach astronomicznych, lecz dostęp do nich, z uwagi na niewielkie nakłady, jest utrudniony. Część danych zawartych w rocznikach można jednak obliczyć samemu. Szybki rozwój techniki obliczeniowej pozwala te skomplikowane rachunki przeprowadzić nawet na mikrokomputerach klasy ZX Spectrum. W kilku najbliższych numerach zamieścimy algorytm służący do wyznaczania współrzędnych równikowych Słońca, Księżyca i planet. Ich znajomość pozwoli obliczać m.in. momenty wschodu i zachodu tych ciał, ich azymuty i wysokości, odległości wzajemne itp., dla dowolnej daty i dla dowolnego miejsca na powierzchni Ziemi.

Główną część tego algorytmu opracowali T. C. Van Flandern i K. F. Pulkkinen z US Naval Observatory i opublikowali w czasopiśmie Astrophysical Journal Supplement Series, 41/391/1979. Jest to najdokładniejszy, analityczny i jednorodny algorytm obliczania współrzędnych Słońca, Księżyca i planet. Błąd wyznaczenia pozycji tych ciał niebieskich wynosi ok. 1' w otoczeniu 300 lat od daty obecnej. Dokładniejsze współrzędne można uzyskać całkując odpowiednie równania ruchu.

Podobnie jak w wielu innych algorytmach astronomicznych potrzebna jest tu znajomość daty juliańskiej. Data juliańska jest to tzw. ciągła rachuba dni. Za dzień juliański o nr 1 (JD = 1) obrano 1 stycznia 4713 r. p.n.e. (zapis algorytmiczny 1 styczeń — 4712 r.), a np. dzień 28 czerwca 1969 roku godz. 0 odpowiada JD = = 2440400.5. Początek daty juliańskiej przypada w południe, stąd część dziesiętna .5.

Poniższy algorytm przedstawia zmianę daty kalendarza gregoriańskiego (obowiązującego od 15 października 1582 roku) na dzień juliański (JD). Jeśli liczbę roku oznaczymy przez R, liczbę miesiąca przez M, a liczbę dnia przez D, to dla daty późniejszej niż 15 X 1582 r. (interesujący nas okres czasu) JD obliczamy w następujący sposób: dla stycznia i lutego obniżamy liczbę roku o 1, a liczbę miesięcy powiększamy o 12, co odpowiada zapisowi algorytmicznemu:

R = R − 1
M = M + 12

dla pozostałych miesięcy R i M zostawiamy bez zmian.

Następnie, za pomocą funkcji INT (stanowiącej część całkowitą liczby, tak że np. INT(3.14) = 3, INT(3.7) = 3, INT(− 3.14) = − 3, INT(− 3.7) = − 3, obliczamy kolejno:

A = INT (R/100)
B = 2 − A + INT (A/4)
JD = INT(365.25*R) + INT(30.6001*(M + 1)) + B + D + 17720994.5

— gdzie JD jest szukaną datą juliańską.

Zasadnicza część algorytmu rozpoczyna się od obliczenia wartości współczynników J1, J2 do J33 (tabela 1).

Tabela 1

J1 = 0.606434+0.03660110129*T1	J1 = −0.29454
J2 = 0.374897+0.03629164709*T1	J2 = −0.07736
J3 = 0.259091+0.03674819520*T1	J3 = −0.28117
J4 = 0.827362+0.03386319198*T1	J4 = −0.56098
J5 = 0.347343−0.00014709391*T1	J5 = −0.01337
J7 = 0.779072+0.00273790931*T1	J7 = −0.73356
J8 = 0.993126+0.00273777850*T1	J8 = −0.51805
J9 = 0.700695+0.01136771400*T1	J9 = −0.98679
J10 = 0.485541+0.01136759566*T1	J10 = −0.20063
J11 = 0.566441+0.01136762384*T1	J11 = −0.12004
J12 = 0.505498+0.00445046867*T1	J12 = −0.09275
J13 = 0.140023+0.00445036173*T1	J13 = −0.45703
J14 = 0.292498+0.00445040017*T1	J14 = −0.30499
J15 = 0.987353+0.00145575328*T1	J15 = −0.23629
J16 = 0.053856+0.00145561327*T1	J16 = −0.16823
J17 = 0.849694+0.00145569465*T1	J17 = −0.37329
J18 = 0.089608+0.00023080893*T1	J18 = −0.48264
J19 = 0.056531+0.00023080893*T1	J19 = −0.51572
J20 = 0.814794+0.00023080893*T1	J20 = −0.75746
J21 = 0.133295+0.00009294371*T1	J21 = −0.90252
J22 = 0.882987+0.00009294371*T1	J22 = −0.15282
J23 = 0.821218+0.00009294371*T1	J23 = −0.21459
J24 = 0.870169+0.00003269438*T1	J24 = +0.50581
J25 = 0.400589+0.00003269438*T1	J25 = +0.03623
J26 = 0.664614+0.00003265562*T1	J26 = +0.30068
J27 = 0.846912+0.00001672092*T1	J27 = +0.66057
J28 = 0.725368+0.00001672092*T1	J28 = +0.53902
J29 = 0.480856+0.00001663715*T1	J29 = +0.29544
J31 = 0.663854+0.00001115482*T1	J31 = +0.53954
J32 = 0.041020+0.00001104864*T1	J32 = −0.08211
J33 = 0.357355+0.00001104864*T1	J33 = +0.23422

Z obliczonych współczynników J1 do J33 (nie ma współczynników J6 i J3) odrzucamy część całkowitą, a zostawiamy do dalszych obliczeń część dziesiętną. Wartości współczynników umieszczone po prawej stronie w tabeli 1 odnoszą się do daty z naszego przykładu tj. dla 28 czerwca 1969 r. godz. 0, gdzie T1 = JD − 2451545, w naszym przykładzie T1 = − 11144.5.

Wprowadźmy teraz dodatkowe wielkości

T = T1/36525 + 1
PI = 3.14159265359
S = PI/180

Obliczenie wartości DL, DB, DR.

Tabela 2 przedstawia współczynniki do obliczania wartości DL, DB i DR dla Słońca.

Tabela 2

Liczba	T	Funkcja	J1	J5	J7	J8	J13	J16	J19
DL									
6910	0	SIN	0	0	0	1	0	0	0
72	0	SIN	0	0	0	2	0	0	0
-17	1	SIN	0	0	0	1	0	0	0
-7	0	COS	0	0	0	1	0	0	-1
6	0	SIN	1	0	-1	0	0	0	0
5	0	SIN	0	0	0	4	0	-8	3
-5	0	COS	0	0	0	2	-2	0	0
-4	0	SIN	0	0	0	1	-1	0	0
4	0	COS	0	0	0	4	0	-8	3
3	0	SIN	0	0	0	2	-2	0	0
-3	0	SIN	0	0	0	0	0	0	1
-3	0	SIN	0	0	0	2	0	0	-2
DB									
0	0	SIN	0	0	0	0	0	0	0
DR									
1.00014	0	COS	0	0	0	0	0	0	0
-0.01675	0	COS	0	0	0	1	0	0	0
0.00014	0	COS	0	0	0	2	0	0	0

Z części tej tabeli obliczamy DL dla Słońca w następujący sposób:

DL = 6910*SIN(J8) + 72*SIN(2*J8)
− 17*T*SIN(J8) − 7*COS(J8 − J18) + ...

(podano przykładowo 4 pierwsze wyrażenia).

Cyfra w kolumnie T oznacza, do której potęgi podnieść T, tj. cyfra 0 oznacza $T^0 = 1$, cyfra 1 oznacza $T^1 = T$ itd. Zauważmy, że opuszczamy współczynniki J, dla których w odpowiednich kolumnach czynniki wynoszą 0. Podobnie wyznaczamy DB i DR z odpowiednich części tabeli. Analogicznie postępujemy z tabelą 3 opisującą współczynniki do obliczania wartości DL, DB i DR dla Księżyca.

Tabela 3

Liczba	T	Funkcja	J2	J3	J4	J5	J7	J8	J12
DL									
22640	0	SIN	1	0	0	0	0	0	0
-4586	0	SIN	1	0	-2	0	0	0	0
2370	0	SIN	0	0	2	0	0	0	0
769	0	SIN	2	0	0	0	0	0	0
-668	0	SIN	0	0	0	0	0	1	0
-412	0	SIN	0	2	0	0	0	0	0
-212	0	SIN	2	0	-2	0	0	0	0
-206	0	SIN	1	0	-2	0	0	1	0
192	0	SIN	1	0	2	0	0	0	0
165	0	SIN	0	0	2	0	0	-1	0
148	0	SIN	1	0	0	0	0	-1	0
-125	0	SIN	0	0	1	0	0	0	0
-110	0	SIN	1	0	0	0	0	1	0
-55	0	SIN	0	2	-2	0	0	0	0
-45	0	SIN	1	2	0	0	0	0	0
40	0	SIN	1	-2	0	0	0	0	0
-38	0	SIN	1	0	-4	0	0	0	0
36	0	SIN	3	0	0	0	0	0	0
-31	0	SIN	2	0	-4	0	0	0	0
28	0	SIN	1	0	-2	0	0	-1	0
-24	0	SIN	0	0	2	0	0	1	0
19	0	SIN	1	0	-1	0	0	0	0
18	0	SIN	0	0	1	0	0	1	0
15	0	SIN	1	0	2	0	0	-1	0
14	0	SIN	2	0	2	0	0	0	0
14	0	SIN	0	0	4	0	0	0	0
-13	0	SIN	3	0	-2	0	0	0	0
-11	0	SIN	1	0	0	0	16	0	-18
10	0	SIN	2	0	0	0	0	-1	0
9	0	SIN	1	-2	-2	0	0	0	0
9	0	COS	1	0	0	0	16	0	-18
-9	0	SIN	2	0	-2	0	0	1	0
-8	0	SIN	1	0	1	0	0	0	0
8	0	SIN	0	0	2	0	0	-2	0
-8	0	SIN	2	0	0	0	0	1	0
-7	0	SIN	0	0	0	0	0	2	0
-7	0	SIN	1	0	-2	0	0	2	0
7	0	SIN	0	0	0	1	0	0	0
-6	0	SIN	1	-2	2	0	0	0	0
-6	0	SIN	0	2	2	0	0	0	0
-4	0	SIN	1	0	-4	0	0	1	0
4	1	COS	1	0	0	0	16	0	-18
-4	0	SIN	2	2	0	0	0	0	0
4	1	SIN	1	0	0	0	16	0	-18
3	0	SIN	1	0	-3	0	0	0	0
-3	0	SIN	1	0	2	0	0	1	0
-3	0	SIN	2	0	-4	0	0	1	0
3	0	SIN	1	0	0	0	0	-2	0
3	0	SIN	1	0	-2	0	0	-2	0
-2	0	SIN	2	0	-2	0	0	-1	0
-2	0	SIN	0	2	-2	0	0	1	0
2	0	SIN	1	0	4	0	0	0	0
2	0	SIN	4	0	0	0	0	0	0
2	0	SIN	0	0	4	0	0	-1	0
2	0	SIN	2	0	-1	0	0	0	0
DB									
18461	0	SIN	0	1	0	0	0	0	0
1010	0	SIN	1	1	0	0	0	0	0
1000	0	SIN	1	-1	0	0	0	0	0
-624	0	SIN	0	1	-2	0	0	0	0
-199	0	SIN	1	-1	-2	0	0	0	0
-167	0	SIN	1	1	-2	0	0	0	0
117	0	SIN	0	1	2	0	0	0	0
62	0	SIN	2	1	0	0	0	0	0
33	0	SIN	1	-1	2	0	0	0	0
32	0	SIN	2	-1	0	0	0	0	0
-30	0	SIN	0	1	-2	0	0	1	0
-16	0	SIN	2	1	-2	0	0	0	0
15	0	SIN	1	1	2	0	0	0	0
12	0	SIN	0	1	-2	0	0	-1	0
-9	0	SIN	1	-1	-2	0	0	1	0
-8	0	SIN	0	1	0	1	0	0	0
8	0	SIN	0	1	2	0	0	-1	0
-7	0	SIN	1	1	-2	0	0	1	0
7	0	SIN	1	1	0	0	0	-1	0



-7	0	SIN	1	1	-4	0	0	0	0
-6	0	SIN	0	1	0	0	0	1	0
-6	0	SIN	0	3	0	0	0	0	0
6	0	SIN	1	-1	0	0	0	-1	0
-5	0	SIN	0	1	1	0	0	0	0
-5	0	SIN	1	1	0	0	0	1	0
-5	0	SIN	1	-1	0	0	0	1	0
5	0	SIN	0	1	0	0	0	-1	0
5	0	SIN	0	1	-1	0	0	0	0
4	0	SIN	3	1	0	0	0	0	0
-4	0	SIN	0	1	-4	0	0	0	0
-3	0	SIN	1	-1	-4	0	0	0	0
3	0	SIN	1	-3	0	0	0	0	0
-2	0	SIN	2	-1	-4	0	0	0	0
-2	0	SIN	0	3	-2	0	0	0	0
2	0	SIN	2	-1	2	0	0	0	0
2	0	SIN	1	-1	2	0	0	-1	0
2	0	SIN	2	-1	-2	0	0	0	0
2	0	SIN	3	-1	0	0	0	0	0
DR									
60.36298	0	COS	0	0	0	0	0	0	0
-3.27746	0	COS	1	0	0	0	0	0	0
-0.57994	0	COS	1	0	-2	0	0	0	0
-0.46357	0	COS	0	0	2	0	0	0	0
-0.08904	0	COS	2	0	0	0	0	0	0
0.03865	0	COS	2	0	-2	0	0	0	0
-0.03237	0	COS	0	0	2	0	0	-1	0
-0.02688	0	COS	1	0	2	0	0	0	0
-0.02358	0	COS	1	0	-2	0	0	1	0
-0.02030	0	COS	1	0	0	0	0	-1	0
0.01719	0	COS	0	0	1	0	0	0	0
0.01671	0	COS	1	0	0	0	0	1	0
0.01247	0	COS	1	-2	0	0	0	0	0
0.00704	0	COS	0	0	0	0	0	1	0
0.00529	0	COS	0	0	2	0	0	1	0
-0.00524	0	COS	1	0	-4	0	0	0	0
0.00398	0	COS	1	0	-2	0	0	-1	0
-0.00366	0	COS	3	0	0	0	0	0	0
-0.00295	0	COS	2	0	-4	0	0	0	0
-0.00263	0	COS	0	0	1	0	0	1	0
0.00249	0	COS	3	0	-2	0	0	0	0
-0.00221	0	COS	1	0	2	0	0	-1	0
0.00185	0	COS	0	2	-2	0	0	0	0
-0.00161	0	COS	0	0	2	0	0	-2	0
0.00147	0	COS	1	2	-2	0	0	0	0
-0.00142	0	COS	0	0	4	0	0	0	0
0.00139	0	COS	2	0	-2	0	0	1	0
-0.00118	0	COS	1	0	-4	0	0	1	0
-0.00116	0	COS	2	0	2	0	0	0	0
-0.00110	0	COS	2	0	0	0	0	-1	0

Do obliczenia DL, DB i DR dla Słońca i Księżyca wykorzystujemy tylko część współczynników J, pozostałe będą przydatne do wyznaczenia analogicznych wielkości dla planet.

Dla rozpatrywanej daty wartości DL, DB i DR dla Słońca i Księżyca wynoszą:

	DL	DB	DR
Słońce	+749.6	0	1.01665
Księżyc	−14572.6	−17465.7	56.5554

Obliczanie rektascensji α, deklinacji δ i odległości DR od Ziemi.

Obliczamy kolejno:
T2 = (JD − 2415020)/36525
EP = 23.452294 − 0.0130125*T2 − 1.64*10−6*T2*T2 + 5.03*10−7*T2*T2*T2

Dla Słońca
LR = (DL/3600)*S + 2*PI*J7
Dla Księżyca
LR = (DL/3600)*S + 2*PI*J1,
gdzie DL bierzemy odpowiednio obliczone dla Słońca i Księżyca. W obu przypadkach dodajemy do LR poprawkę DLR
DLR = −17*SIN(J5*2*PI)/3600*S
Gdy LR jest ujemne, to dodajemy do niego 2*PI (LR jest wyrażone w radianach).

Dla Słońca i Księżyca obliczamy kolejno:
DBR = (DB/3600)*S
EPR = EP*S
X1 = DR*COS(DBR)*COS(LR)
X2 = DR*COS(DBR)*SIN(LR)
X3 = DR*SIN(DBR)
X4 = X1
X5 = X2*COS(EPR) − X3*SIN(EPR)
X6 = X2*SIN(EPR) + X3*COS(EPR)

Obliczanie rektascensji α
$\alpha = \mathrm{arctg}(X5/X4)/S$,
gdy $X5 < 0$ i $X4 > 0$ to $\alpha = \alpha + 360$
gdy $X5 < 0$ i $X4 < 0$ to $\alpha = \alpha + 180$
gdy $\alpha < 0$ to $\alpha = \alpha + 180$
$\alpha = \alpha/15$, tj. będzie wyrażone w godzinach i w ułamkach godziny. Ułamek ten możemy zamienić na minuty i sekundy.

Obliczanie deklinacji δ
$\delta = \mathrm{arctg}(X6/(X4*X4 + X5*X5)^{1/2})/S$
Deklinacja wyrażona jest w stopniach i zawiera się w przedziale (−90°, +90°).

Wartość DR już wyznaczyliśmy wcześniej. Dla Słońca oznacza ona odległość Ziemia—Słońce wyrażoną w jednostkach astronomicznych (1 j.a. = 149.6 mln km), natomiast DR dla Księżyca oznacza odległość Ziemia—Księżyc, która jest wyrażona w promieniach Ziemi (1 j.a. = 23454.8 promieni Ziemi). Porównanie obliczonych współrzędnych (TEST) z podanymi w Roczniku Astronomicznym (ROCZNIK)

Słońce

	TEST	ROCZNIK
LR/S	96° 07′ 38″	96° 07′ 38″
DBR/S	0° 0′ 0″	−0° 0′ 1″
DR	1.01665	1.01659
α	$6^h 26^m 41^s$	$6^h 26^m 40^s$
δ	+23° 18′ 11″	+23° 18′ 14″

Księżyc

	TEST	ROCZNIK
LR/S	249° 55′ 04″	249° 55′ 03″
DBR/S	−4° 51′ 06″	−4° 51′ 09″
DR	56.55545	56.55235
α	$16^h 29^m 54^s$	$16^h 29^m 54^s$
δ	−26° 44′ 16″	−26° 44′ 27″

Opracował: Ireneusz Włodarczyk

CIEKAWE KSIĄŻKI

Edward Kącki: ELEKTRONICZNE MASZYNY LICZĄCE. Instytut Wydawniczy „Nasza Księgarnia", Warszawa 1988, wydanie pierwsze, nakład 50 000 egz., cena zł 260.

Recenzowana książeczka wydana została w serii **BIBLIOTEKI MŁODEGO TECHNIKA**, co jednoznacznie określa czytelnika, do którego jest adresowana. Można przyjąć, że jest to człowiek w wieku od kilkunastu do dwudziestu kilku lat, niekoniecznie z wykształceniem technicznym, a panujący u nas boom mikrokomputerowy pozwala nawet przesunąć dolną granicę wiekową nieco niżej. Niniejsza recenzja pisana jest przy założeniu, że odbiorcą tej książki jest taki właśnie młodzieniec.

Po przeczytaniu książki odniosłem wrażenie, że celem Autora było napisanie czegoś w rodzaju kompendium wiedzy na temat wszelkiego rodzaju maszyn liczących, ich konstrukcji i zastosowania. Zamiar to chwalebny i ze wszech miar godny poparcia, tyle że, niestety, niewykonalny w książeczce o wymiarach 105 × 160 mm liczącej ok. 120 stron czystego tekstu. Trzeba jednak oddać Autorowi sprawiedliwość, że robił, co mógł. Bardzo niekorzystnie odbiło się to na stylu książki — jest ona pisana językiem suchym, hasłowym i przeładowana informacjami o (według mnie) znikomym stopniu praktycznej użyteczności. Uważam, że nieco „gadulstwa" i humoru znacznie by ułatwiło odbieralność treści książki młodemu czytelnikowi. Pamiętajmy, że ideę tę wyznawali wielcy naszej literatury, jak np. Julian Tuwim (baw ucząc).

Informacja zawarta w książce ma charakter rzetelny, chociaż niekiedy przestarzały. Odnosi się wrażenie, że ostatnie informacje o charakterze technicznym zamieszczone w książce pochodzą z końca lat siedemdziesiątych; mamy obecnie rok 1989, a ostatnia dekada była przełomowa w dziedzinie rozwoju sprzętu obliczeniowego i technologii związanych z jego wytwarzaniem. Nie znalazło to odbicia w treści, co jest tym dziwniejsze, że książkę oddano do produkcji w 1986 roku.

Treść książki podzielić można umownie na trzy części: króciutki wstęp („Co to jest informatyka?"), cztery rozdziały mające przybliżyć czytelnikowi problemy podziału, konstrukcji i zasad pracy z maszynami liczącymi, a następnie pięć rozdziałów o wykorzystaniu maszyn liczących w różnych dziedzinach życia, a mianowicie: technice, zarządzaniu i administracji, medycynie, procesach tworzenia sztucznej inteligencji i wojskowości. Przyznam się, że podział ten wydał mi się wydumany, gdyż wszystkie te działy zazębiają się i przenikają. Taka skutecznie zaciemniająca obraz sprawy klasyfikacja jest absolutnie zbędna w książce popularnej!

Autor konsekwentnie nie używa słowa *komputer* posługując się wyrażeniami *maszyna matematyczna* (zalecane przez **Encyklopedię Techniki**) lub *maszyna licząca*. Wydaje się, że wyraz *komputer* na tyle zadomowił się w naszym języku, że jest to nadmiar dbałości o czystość językową (ale chyba wybaczalny).

Wprowadzona przez Autora (bardzo rozbudowana) klasyfikacja wykazuje pewne nieścisłości (np. maszyna siatkowa i układ siatkowy).

Z trzech używanych u nas (nie wszystkich formalnie) symboli oznaczających funktory logiczne Autor wybrał najmniej popularny, co skutecznie zmniejsza wyrazistość przykładów (o ile pominiemy błędy rysunkowe, np. w schemacie sumatora na str. 52).

Dyskusyjny jest również dobór przykładów. Wydaje się, że równanie Laplace'a stanowi przykład trochę za wysokich lotów dla odbiorcy książki, który nie będzie w stanie docenić głębokości wiedzy eksponowanej przez Autora ze względu na brak odpowiedniego przygotowania.

Podsumowując walory merytoryczne pozycji: jeżeli potraktujemy ją jako wstępny konspekt do ogromnego (być może wielotomowego) dzieła, ocena będzie bardzo wysoka, jednak próba wydania jej jako kolejnego tomu **BIBLIOTEKI MŁODEGO TECHNIKA** jest przedsięwzięciem absolutnie chybionym. Dodatkową wadą jest bardzo oszczędny spis literatury, tak że czytelnik z niedosytem wiedzy nie bardzo będzie wiedział, gdzie ją może pogłębić.

„Gwoździem do książki" niech będzie sposób jej wydania. Przeczytanie książki spowodowało, że rozpadła się ona na szereg niczym nie połączonych kawałków (w tym wiele pojedynczych kartek) i okładkę. Miało to i swoje dobre strony: podczas tworzenia tej recenzji mogłem, w miarę potrzeb, wykorzystywane fragmenty układać obok *elektronicznej maszyny liczącej*, za pomocą której pisałem ten tekst, bez obawy, że mi się zamkną.

(dap)

Cz. Kosniowski — ZANIMATIELNAJA MATEMATIKA I PERSONALNYJ KOMPUTER, wydawnictwo „Mir", Moskwa 1987.

Publikacja ta jest rosyjskim tłumaczeniem wydanej w 1984 roku książki Czesa Kosniowskiego „Fun mathematics on your microcomputer". Przeznaczona jest dla osób piszących swe pierwsze programy w języku BASIC, a jednocześnie interesujących się matematyką. Autor w bardzo przystępny sposób omawia krótko poszczególne działy matematyki, a następnie przedstawia gotowe programy w języku BASIC, będące „komputerową ilustracją" przekazywanych wiadomości. Prawie każdy przedstawiony program jest formą zabawy z komputerem, a nie „suchym" przedstawieniem teorii i definicji matematycznych, dlatego też gorąco polecałbym tę książkę nauczycielom matematyki, mającym w szkole (często nie wykorzystywany z braku odpowiednich programów edukacyjnych) komputer, jak również młodzieżowym klubom komputerowym. Każdy program zawiera obszerne komentarze i jest napisany za pomocą instrukcji, występujących praktycznie we wszystkich dialektach BASIC-a. Fragmenty programu zawierające instrukcje charakterystyczne dla danego typu komputera (np. „czyszczenie" ekranu i wyprowadzanie wyników na ekran) autor przedstawia w kilku wersjach. Niezależnie od tego na końcu książki znajdziemy dodatek zawierający praktyczne wskazówki dotyczące przeróbki przedstawionych programów na różne dialekty BASIC-a (np. Commodore, BBC, ZX81, Spectrum), co czyni książkę bardzo uniwersalną i przydatną posiadaczom różnych mikrokomputerów.

Przytoczmy słowa samego Autora — „Z książki tej dowiecie się jak interesujące może być poznawanie wspaniałego świata matematyki z pomocą mikrokomputera". I myślę, że Autor cel ten osiągnął.

Poszczególne rozdziały książki omawiają między innymi: ciągi i szeregi, wykresy funkcji w układzie współrzędnych kartezjańskich i biegunowych, macierze, teorię gier, teorię grup oraz równania różniczkowe. Oczywiście teoria „wykładana" jest na poziomie ucznia szkoły średniej. Poza tym każdy rozdział stanowi niezależną całość i może być wykorzystany oddzielnie bez potrzeby czytania całej książki. Zaletą tej publikacji jest również dwukolorowy druk — listingi programów, rysunki oraz „ekran monitora" drukowane są w kolorze niebieskim, co uprzyjemnia czytanie tej lektury.

Uważam, że przedstawiona książka będzie również przydatna wszystkim uczącym się języka BASIC. Przedstawione pomysły można odpowiednio „rozwinąć", a następnie wykorzystać na lekcjach matematyki. Z pewnością uczyni je bardziej atrakcyjnymi dla uczniów. Na zakończenie przytoczmy jeszcze słowa Autora — „Książka ta nauczy was wielu pożytecznych w informatyce i matematyce rzeczy. Będzie dla was niewyczerpanym źródłem pomysłów. Wiadomości, które zdobędziecie, pozwolą wam samodzielnie pisać i bardziej złożone programy niż te zamieszczone w książce". Dodatkową zachętą do kupna tej książki niech będzie i to, że jej polska cena (z 1987 r.) wynosi jedynie 120 zł.

Zbigniew Krauze

„Młody Technik — InforMik" wydaje Instytut Wydawniczy „Nasza Księgarnia"

Rada Redakcyjna: doc. dr Zygmunt Dąbrowski, inż. Jerzy Jasiuk, dr Zygmunt Kalisz, mgr Zbigniew Słowiński, mgr inż. Jerzy Siek, dr Zbigniew Płochocki, Piotr Postawka, mgr inż. Roland Wacławek, prof. dr hab. Andrzej K. Wróblewski (przewodniczący), mgr inż. Grzegorz Zalot.

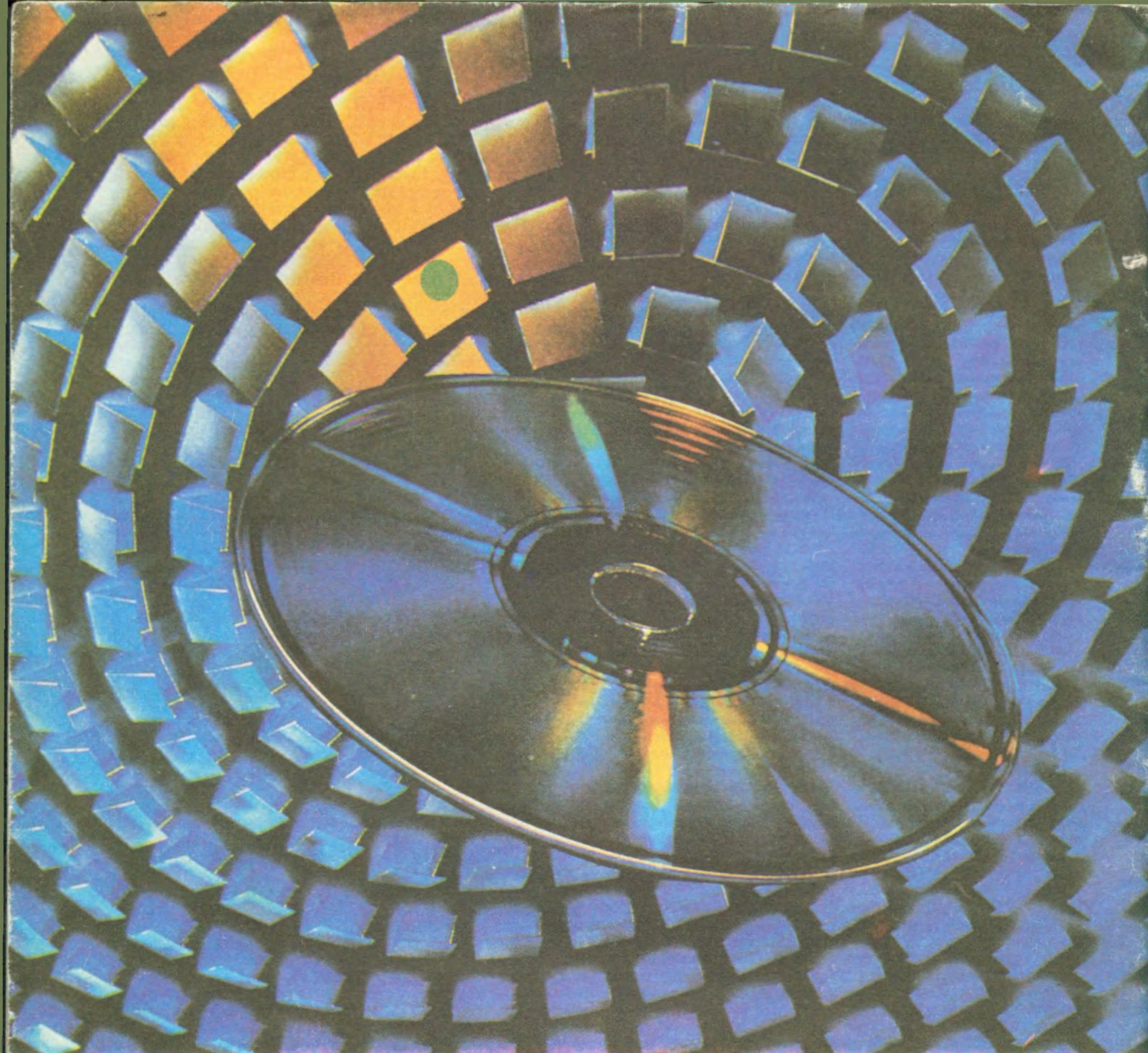
Zespół redakcyjny: *„InforMik"* redaguje zespół *„Młodego Technika"*. Jerzy Klawiński (sekretarz red.), Jacek Nowicki (red.), Dariusz A. Przygoda (red.), Lidia Sadowska-Szlaga (korekta), Józef Trzionka (redaktor naczelny), Roland Wacławek (software), Grzegorz Zalot (hardware), Izabela Żur (red. tech.).

Stali współpracownicy: Wojciech Apel, Tadeusz Basista, Jacek Jędrzejowski, Piotr Postawka, Marek Szczepański, Krzysztof Wiśniewski.

Adres redakcji: ul. Spasowskiego 4, 00-389 Warszawa, lub skr. poczt. 380, 00-950 Warszawa. **Telefony:** centrala: 26 24 31 do 36. Dział Łączności z Czytelnikami — wewn. 60, pozostałe działy: wewn. 42 i 47. Redaktor naczelny: 26 26 27 lub wewn. 87.

Warunki prenumeraty: ogólnie obowiązujące w kraju. **W STAŁEJ SPRZEDAŻY INFORMIK JEST W SALONIE WYDAWNICZYM „NASZEJ KSIĘGARNI"** ul. Spasowskiego 4 A.

Druk: Zakłady Graficzne w Katowicach. Zam. 0599/4333/9
Nakład 100 315 egz. A-19



Dokończenie ze str. 28

strony (rys. 2). Ponieważ laser generuje światło monochromatyczne (o jednej długości fali), ciągłe i praktycznie o jednym kierunku polaryzacji, jest dla tych zastosowań idealnym źródłem światła. Jeżeli na drodze odbitego od nośnika światła umieścimy filtr polaryzacyjny, to do fotodiody typu PIN będzie docierał strumień światła o natężeniu zależnym od zapisanej informacji.

Zmiany fotoprądu diody są już łatwe do wzmocnienia i dalszej obróbki elektronicznej.

Jedną z firm, która bardzo poważnie zainteresowała się pamięciami optycznymi, jest znany z produkcji optyki i aparatów fotograficznych „Olympus". Na jednej dyskietce jego produkcji o wymiarach 5,25 cala mieści się 3400 obrazów komputerowych. Jest to 250 razy więcej, niż na klasycznych dyskach. Jak ocenia „Olympus", początkowo 70 do 80% dysków optycznych nie będzie w prywatnych rękach. Już teraz duże zainteresowanie tym nowym nośnikiem informacji wykazują banki, biblioteki, banki danych farmaceutycznych i kartoteki dużych szpitali. Sam „Olympus" ma zamiar do roku 1990 wydać na ten cel 2,2 miliarda dolarów.

Krzysztof Wiśniewski

Cena zł 200,— Indeks nr 366013 PL ISSN 0860-5696